

Microsoft Access 2007

Domeniul bazelor de date este foarte important la ora actuala. Indiferent ca suntem un utilizator obisnuit, unul experimentat sau un profesionist IT, studiul unei baze de date este mai important ca oricand.

Introducere

O sa incepem cu o baza de date noua, creata de la zero, o sa discutam in mare despre ce reprezinta o baza de data, la ce este folosita ea, s.a.m.d. Apoi o sa incepem sa cream efectiv componentele din interiorul unei baze de date. Este foarte important sa intelegem ce reprezinta aceste componente si la ce ne sunt de fapt ele utile. Vorbim despre tabele, despre constrangeri, despre relatii. As putea spune ca cele mai importante entitati din interiorul unei baze de date sunt:

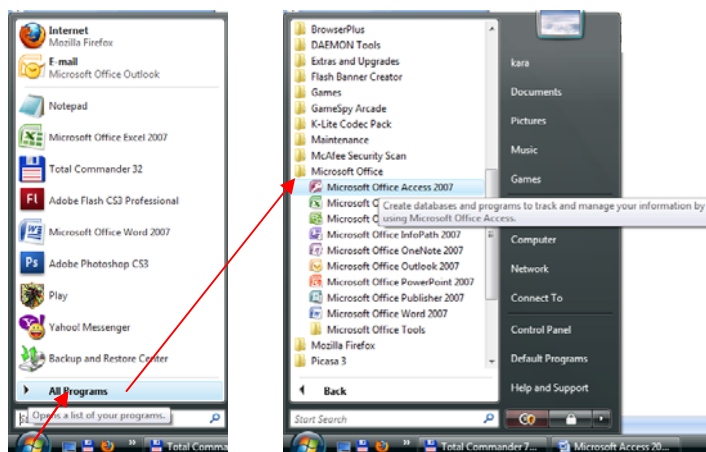
- **Interogariile**, ce ne ajuta sa extragem anumite informatii dintr-o baza de date;
- **Rapoartele**, ce ne permit sa scoatem pe hartie aceste date (intr-un raport conteaza si partea estetica, layout-ul, felul cum sunt prezentate acele date);
- **Formularele**, ce ne ajuta sa introducem sau sa modificam datele din interiorul unei baze de date.

Creare unei baze de date de la 0

In Microsoft Access 2007 avem la indemana foarte multe sabloane de baze de date, toate aceste sabloane contin baze de date deja realizate. In acest exemplu incepem o baza de date de la 0.

O sa incepem sa cream o baza de date noua. Deschidem aplicatia **Microsoft Access 2007**.

Lansarea in executie a programului se face urmand calea: **Start → All Programs → Microsoft Office → Microsoft Office Access 2007**.





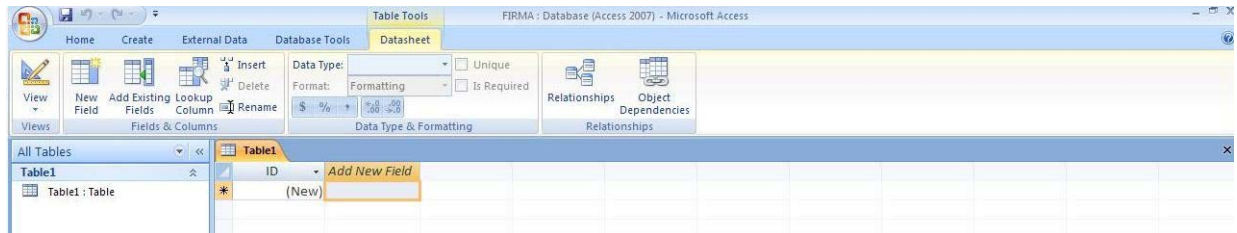
O sa pornim cu o baza de date de la zero. O sa dam un nume acestei baze de date, de exemplu **FIRMA.accdb**.



Apasam **Create**, si in acest moment putem sa cream componentele unei baze de date.

Daca ar fi sa dau o definitie unei baze de date, as spune cat mai simplu, in felul urmator: *o baza de date nu este altceva decat o suma de tabele, tabele care contin date din viata reala, iar aceste tabele, in interiorul bazei de date, sunt relationate*. Ce anume se poate stoca intr-un tabel, si ce anume constituie relatie intre tabele o sa discutam putin mai incolo.

Observati ca interfata **Access**-ului porneste deja cu un tabel nou. Tabelele din interiorul bazei de date seaman foarte mult cu tabele din alte programe cu care poate ati mai lucrat (**Excel**, **Word**), numai ca aceste tabele din **Access** sunt mai ‘inteligente’.



O sa stabilim cateva repere pentru a fi mai usor de inteles urmatoarele tutoriale:

- Orice *coloana* dintr-un tabel se numeste *field*;
- Orice *linie* dintr-un tabel se numeste *record* (*inregistrare*);
- *Intersectia* dintre o linie si o coloana se numeste *value* (*valoare*).

Cea mai importanta linie dintr-un tabel este acea linie care defineste tabelul. Ea se numeste **cap de tabel** (sau *structura de baza*). Acesta este primul lucru pe care il cream atunci cand vrem sa facem un tabel.

Capul de tabel contine definitia coloanelor din tabelul respectiv.

Crearea unui tabel

Tabelele din **Access** sunt mult mai "**inteligente**" decat cele din **Excel**. Au o structura bine definita, fiecare coloana avand un anumit tip de date, o anumita dimensiune, etc.

Este recomandat ca fiecare tabel dintr-o baza de date sa stocheze informatii referitoare la o singura entitate din viata reala.

De exemplu, tabela *Clienti* sa stocheze doar informatii despre clienti, tabela *Produce* doar informatii despre produse, tabela *Tranzactii* doar informatii despre tranzactii, s.a.m.d. Nu stocam in interiorul unei singure tabele informatii amestecate, ce se refera la entitati diferite.

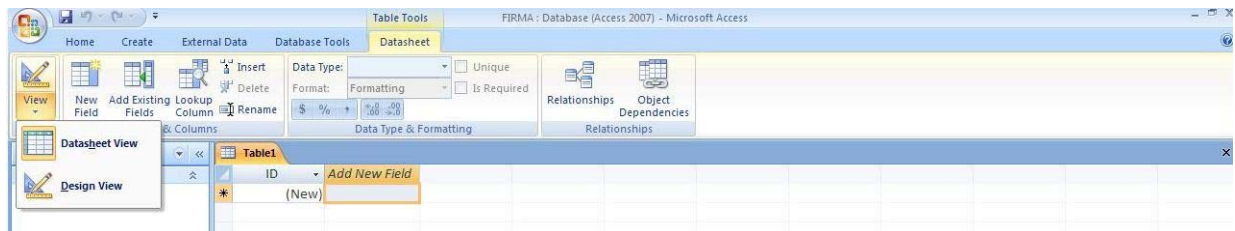
Vom crea un prim tabel numit *Clienti*. Observati ca imediat dupa ce am inceput sa cream baza de date, **Access** se asteapta sa cream capul de tabel al primului meu tabel din baza de date. Cu alte cuvinte, in acest moment putem sa cream structura de baza a tabelului.

Exista doua moduri de vizualizare ale unui tabel (din ribbon-ul contextual **Datasheet > Views**): **Datasheet View**, ce ne permite sa cream structura tabelului si sa introducem datele in acelasi timp, si **Design View**.

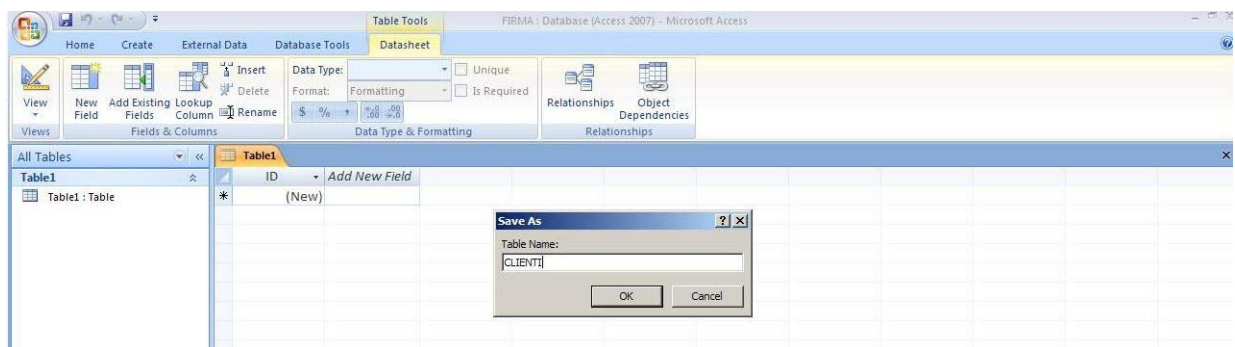
Va recomand sa creati structura tabelului in **Design View**, apoi sa folositi **Datasheet View** pentru introducerea datelor.

Cei care ati mai lucrat in **Excel**, o sa observati ca **Access** este mult mai riguros in ceea ce priveste crearea coloanelor, introducerea datelor, s.a.m.d.

In momentul in care alegem sa cream un tabel in **Design View**, **Access** va cere imediat sa introducem numele acestuia.

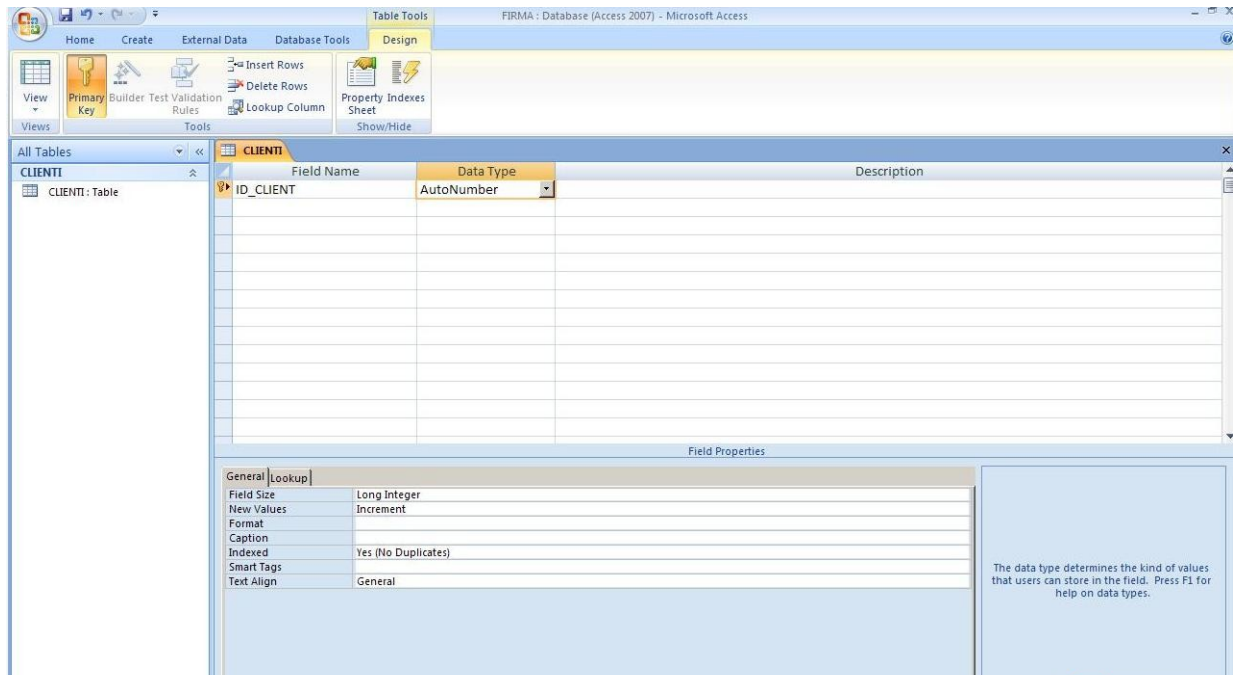


Completam numele *CLIENTI* si apasam **OK**.



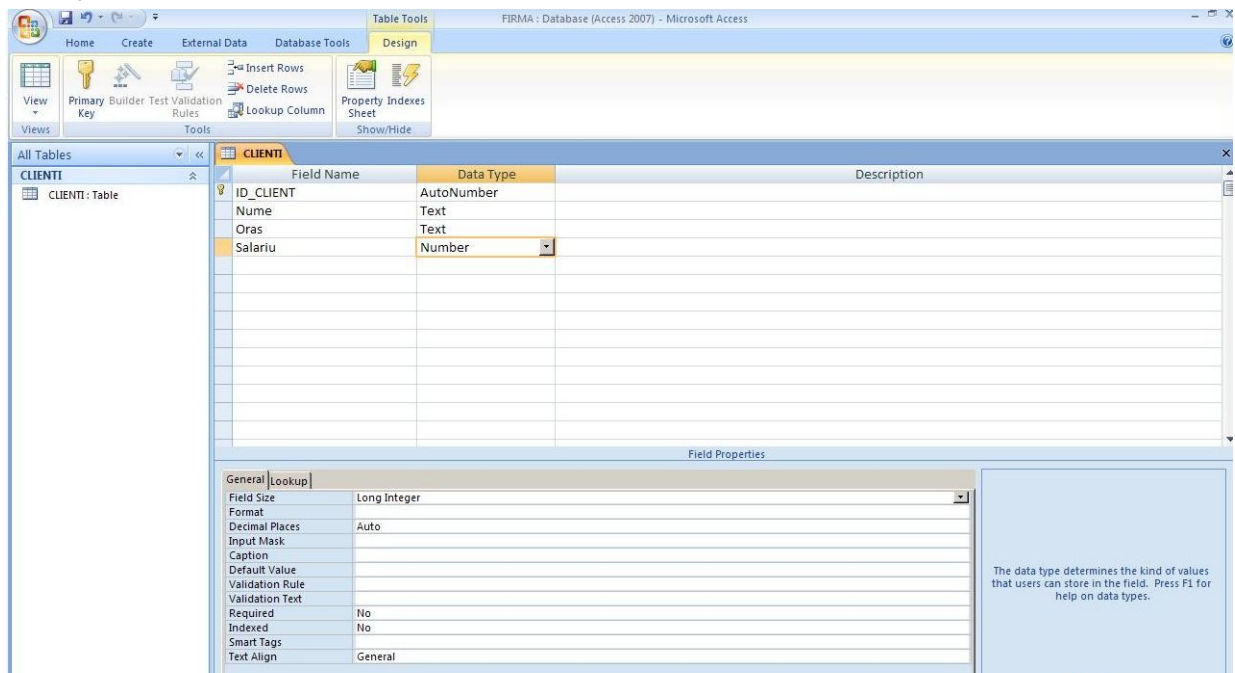
Gasim acum tabela in partea stanga a ecranului, acolo unde avem un meniu ce ne permite sa navigam prin elementele din interiorul bazei de date. Acum putem sa cream coloanele din tabel.

Este recomandat ca orice tabel sa aiba o coloana cu rolul de *identificator unic de linie*. Fie ca se numeste *ID Client* (ca in cazul nostru), CNP sau Marca Client. Daca nu avem un cod intern pentru clienti, numim coloana simplu *ID_CLIENT*, apoi declaram coloana de tip **AutoNumber**. Aceasta inseamna ca **Access** va genera in mod automat un numar de cod unic, sub forma de numar current (1, 2, 3...).



Completam si celelalte coloane, specificand tipul de date pentru fiecare coloana in parte, astfel:

- *Nume*, coloana de tip **Text**;
- *Oras*, coloana de tip **Text**;
- *Salariu*, coloana de tip **Number**; observam ca la tipul **Number** mai exista niste subtipuri de date; alegem **LongInteger**.



In acest moment putem vizualiza tabelul in modul **Datasheet View** (ribbon-ul contextual **Datasheet** > **Views**) pentru a putea introduce inregistrarile.

Apasati **YES** daca Access va cere sa salvati datele.



Observati cum capul de tabel a ramas cu cele patru coloane. In acest moment putem sa introducem cateva linii. Navigarea de la o valoare la alta se poate face apasand tasta **Tab**.

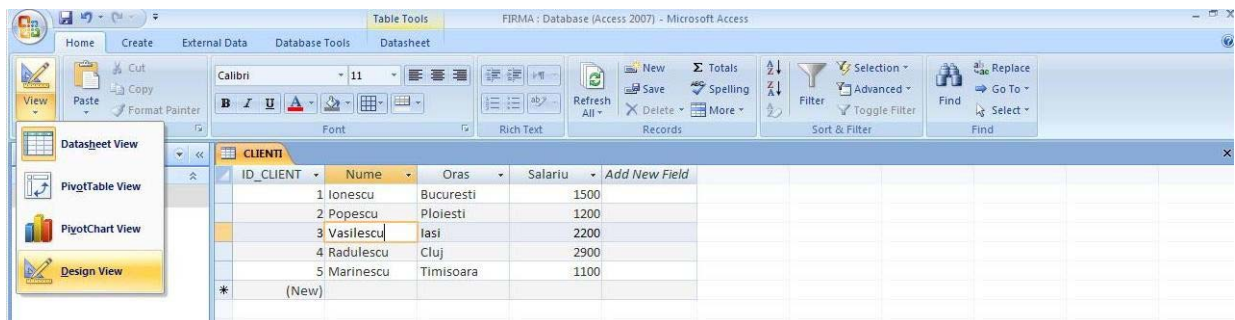
Crearea constrangerilor

Constrangerile sunt niste conditii logice pe care le stabilim pe anumite coloane pentru a **INTEGRITATEA DATELOR** (datele sa fie conforme cu realitatea sau cu regulile de afacere din companie).

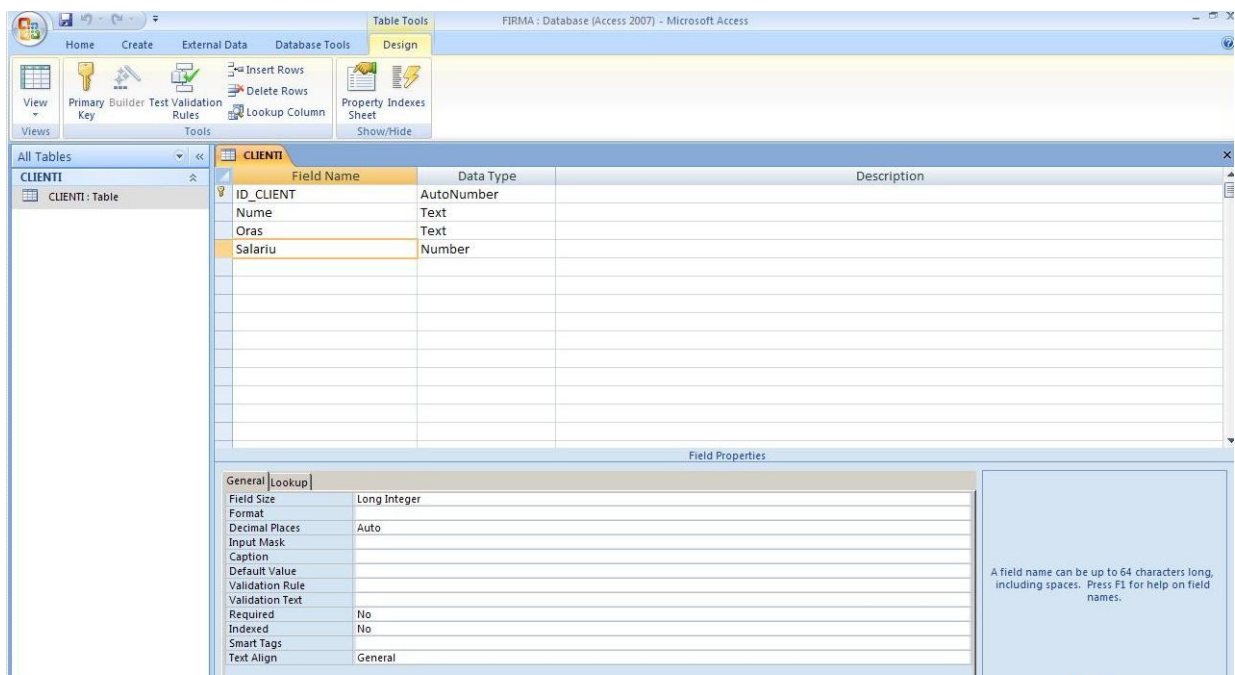
De ex. pe o coloana numita **SALARIU** sa acceptam decat valori mai mari decat salariul minim pe economie.

Constrangerile sunt niste reguli, niste conditii logice, pe care le putem impune coloanelor, in interiorul unor tabele. De exemplu coloana *Salariu* nu are voie sa accepte valori mai mici decat salariul minim pe economie ($\text{Salariu} > 600$).

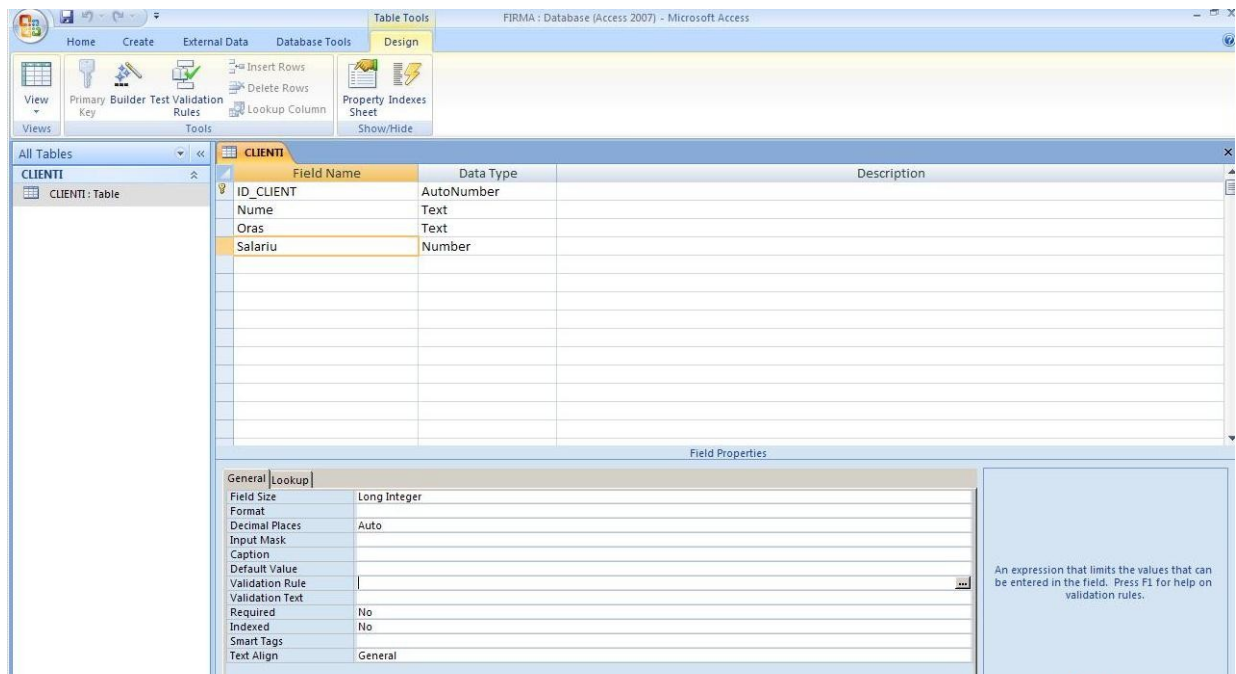
Mergem in modul de vizualizare **Design View**.



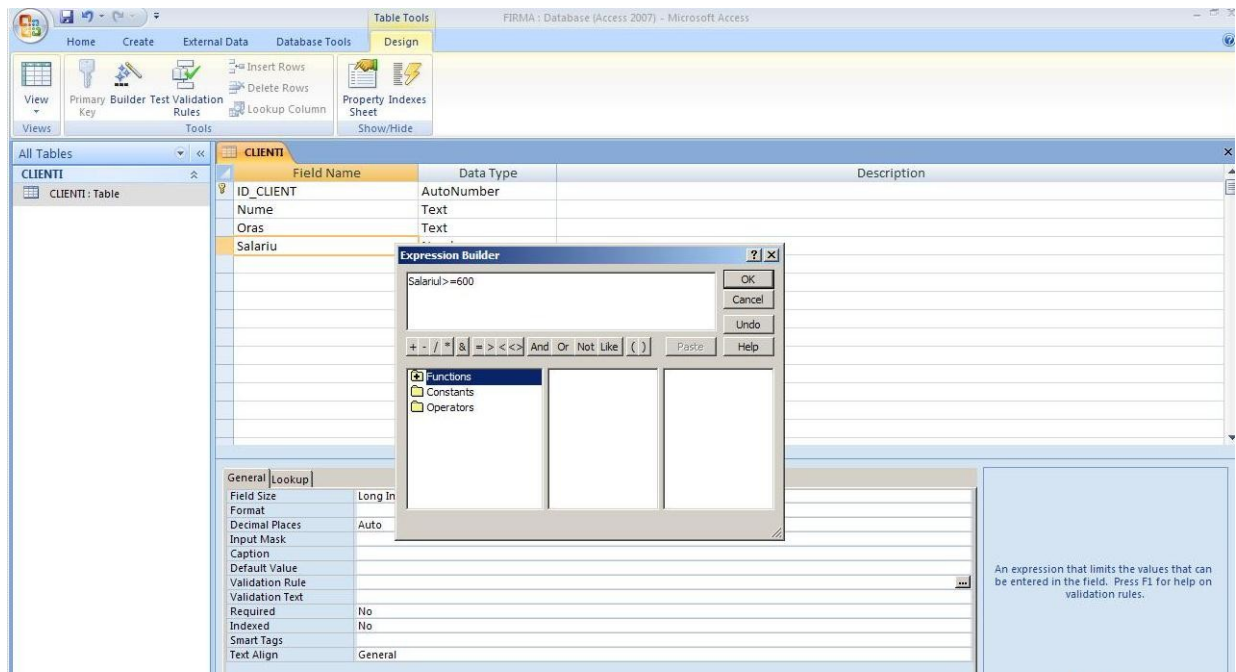
Ne pozitionam pe coloana *Salariu*. Observati ca fiecare coloana are o serie de proprietati – in partea de jos a ecranului – **Field Size**, **Format**, etc.



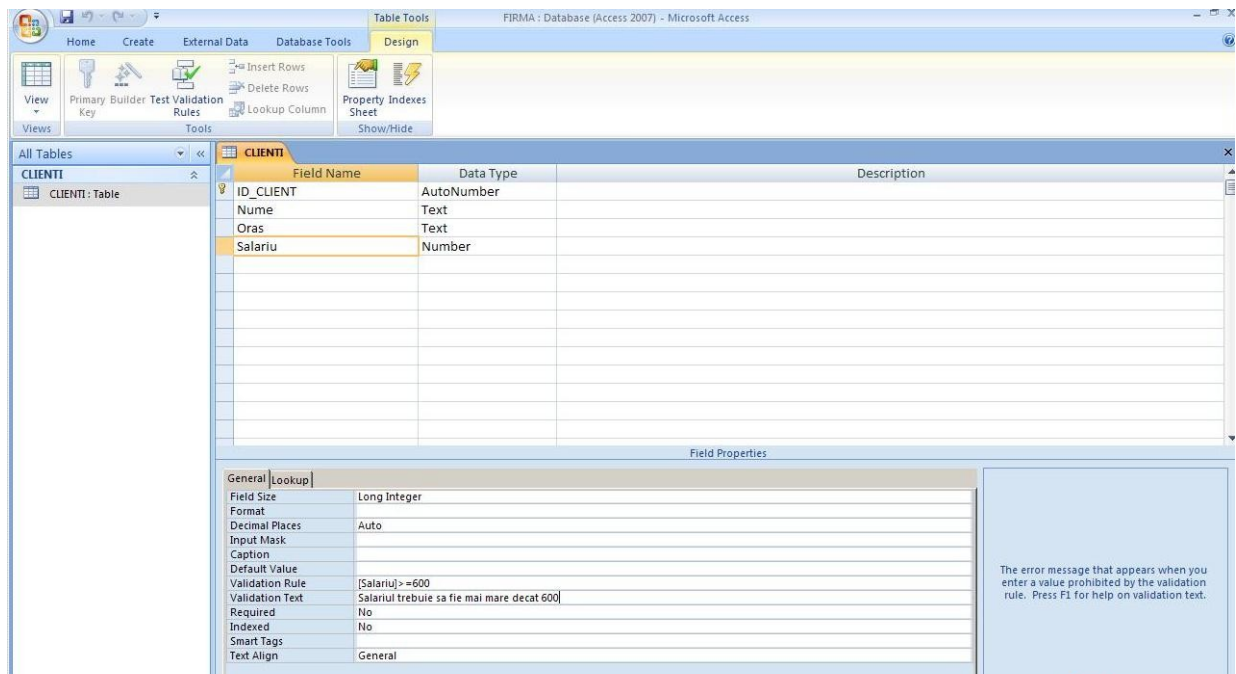
Daca vrem sa stabilim pentru coloana salariu o conditie logica, mergem in campul **Validation Rule** si facem click pe butonul din dreapta campului.



Se deschide o refeastra in care putem sa introducem conditia logica. In primul camp scriem: **Salariul>=600**, si apasam **OK**.

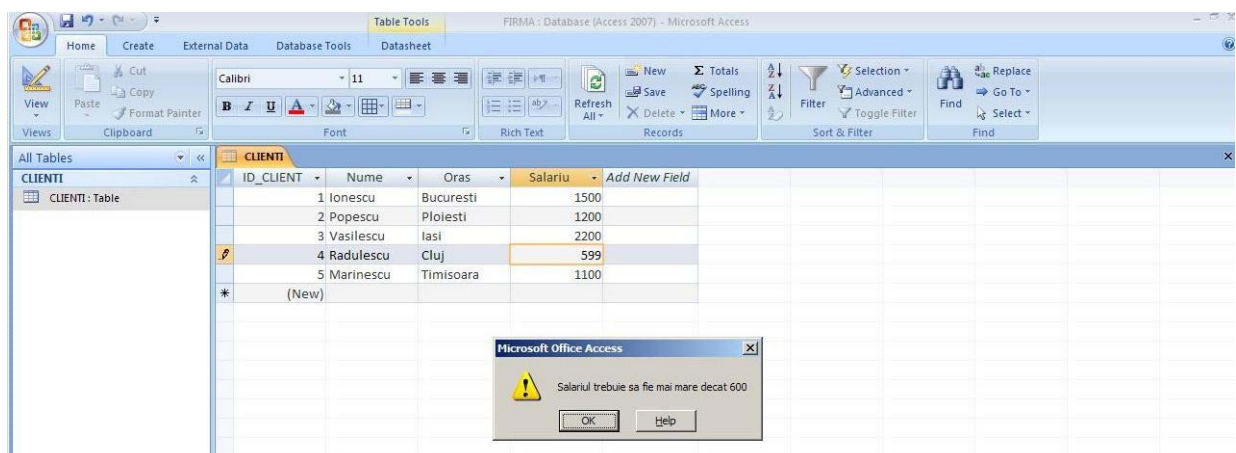


In **Validation Text** vom introduce un mesaj text ce va aparea pe ecran in mod automat atunci cand conditia nu este indeplinita: *Salariul trebuie sa fie mai mare decat 600.*



Salvam modificarile si mergem in modul **Datasheet View**.

Pentru una dintre inregistrari (de exemplu *Radulescu*) modificam salariul la 599. In momentul in care vrem sa salvam sau sa plecam de pe linia respectiva, apare **Validation Text**-ul si nu putem parasii campul atata timp cat conditia mentionata nu este indeplinita.



Putem apasa tasta **Esc** pentru a parasii campul, care va reveni la valoarea anterioara.

Atunci cand modificam valoarea la una mai mare de 600, mesajul nu va mai fi afisat.

Trebuie sa retineti ca exista mai multe tipuri de constrangeri: constrangerea de tipul *Primary Key*, constrangerea de tip *Unique*, constrangerea de tip *Not Null*, constrangerea de tip *Foreign Key*. Nu putem sa vorbim de toate acum, dar este de retinut ca orice conditie pusa pe o anumita coloana rezulta intr-o constrangere. Este recomandat sa folosim constrangeri pentru a fi siguri ca in baza de date nu ajung decat valori conforme cu realitatea.

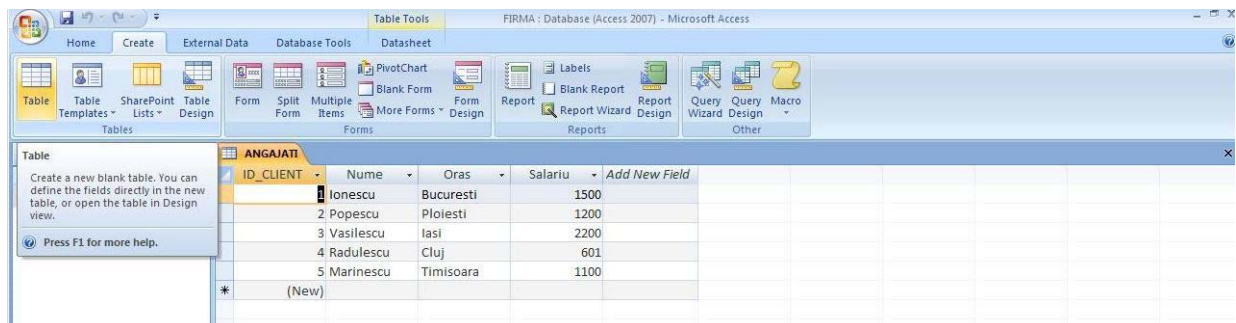
Crearea relatiilor

Fiecare tabel dintr-o baza de date trebuie sa stocheze valori bine definite din viata reala. Pentru entitati diferite din viata reala trebuie sa creem tabele diferite. De ex. : Tabela **Angajati** stocheaza decat informatii specifice angajatilor. Tabela **COPII_ANGAJATI** stocheaza decat informatii specifice copiilor de angajati. Totusi, pentru a se pastra legatura din viata reala dintre entitati (un angajat poate avea mai multi copii, etc), trebuie sa relationam tabelele dintr-o baza de date.

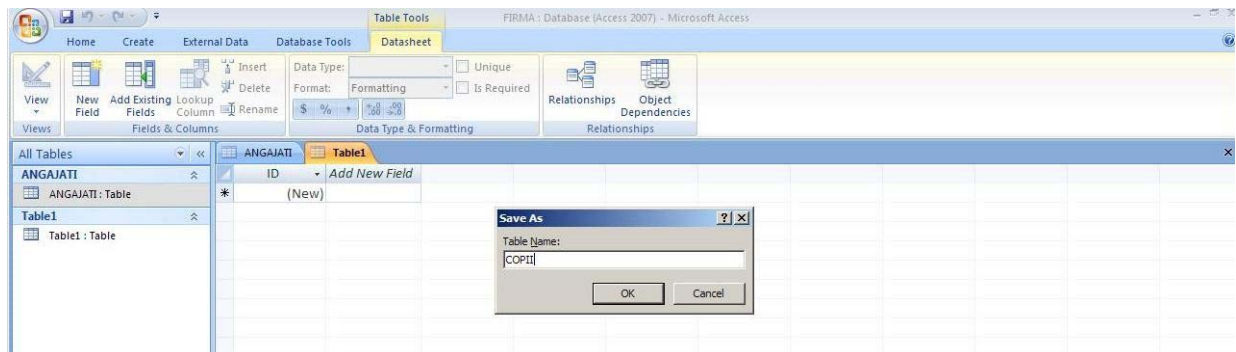
Intr-o baza de date putem avea mai multe tabele. Am convenit deja ca un tabel se refera practic la o singura entitate din viata reala.

In cazul in care avem mai multe tabele, se impune ca aceste tabele sa fie relationate, asa cum entitatile din viata reala sunt relationate. Acest concept legat de relationare sta la baza *teoriei relationale*, care, daca vreti, este inima notiunii unei baze de date. Deosebirea esentiala dintre o baza de date **Access** si un alt software (**Excel**, **Word**) este tocmai aceea ca obiectele, entitatile, tabelele din interiorul bazei de date sunt relationate.

Sa presupunem ca mai avem o tabela unde tinem datele despre copiii angajatilor. Din ribbon-ul **Create** cream un nou tabel (**Table**).

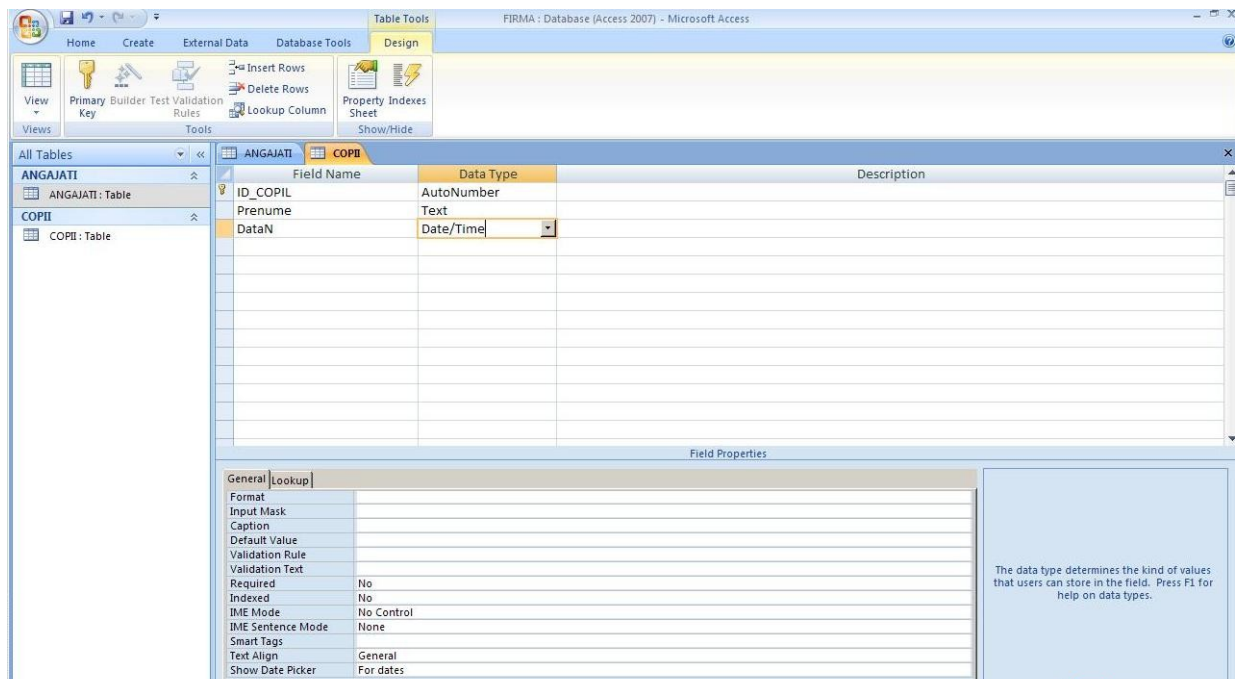


Mergem in modul de vizualizare **Design View**, pentru a realiza structura tabelului. In acest moment introducem numele noului tabel: **COPII**. Apasam **OK**.



Structura tabelului va fi urmatoarea:

- **ID_COPIL**, de tip **AutoNumber**;
- **Prenume**, de tip **Text**;
- **DataN**, de tip **Date/Time**.



Salvam modificarile si mergem in modul **Datasheet View**. Introduceti o serie de date ca in screen-shotul de mai jos.



Felul in care introducem o data calendaristica este conform cu **Regional Settings** din sistemul de operare. Trebuie sa mergem in **Control Panel > Regional and Language Options** si sa vedem ce **Language** avem setat. **Microsoft Access** este dependent de sistemul de operare privind aceste setari de format al datei.

Acum avem doua tabele: **ANGAJATI** si **COPII**. Asa cum in viata reala exista o legatura intre angajati si copiii lor, asa si in **Access** trebuie sa avem acea relationare intre cele doua tabele, intre cele doua entitati separate. Ramane in continuare sa stabilim cum relationam tabelele, adica cum stabilim pentru fiecare angajat ce copil are.

Fiecare tabel dintr-o baza de date trebuie sa stocheze valori bine definite din viata reala. Pentru entitati diferite din viata reala trebuie sa creem tabele diferite. De ex. : Tabela Angajati stocheaza decat informatii specifice angajatilor Tabela COPII_ANGAJATI stocheaza decat informatii specifice copiilor de angajati. Totusi, pentru a se pastra legatura din viata reala dintre entitati (un angajat poate avea mai multi copii, etc), trebuie sa relationam tabelele dintr-o baza de date.

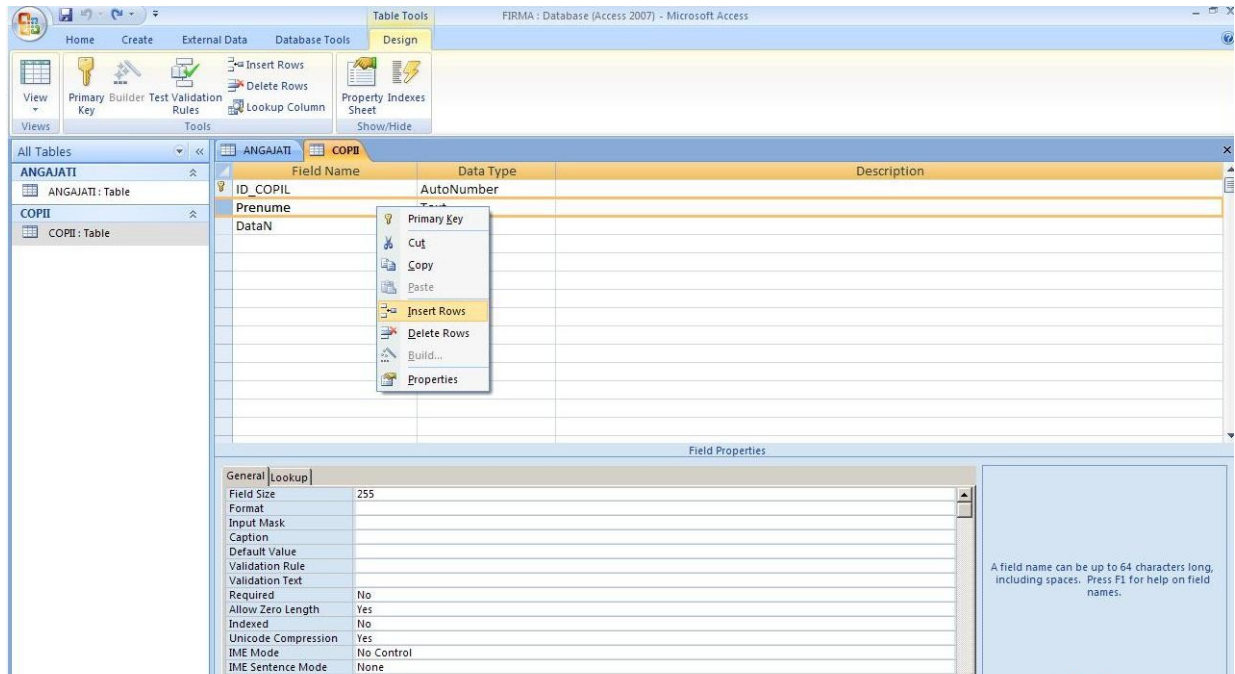
Avem doua tabele: **ANGAJATI** (stocheaza informatii despre angajati) si **COPII** (stocheaza informatii despre copii angajatilor). Asa cum in viata reala unui angajat ii pot corespunde unul sau mai multi copii, trebuie sa stabilim si noi aceasta relatie in interiorul bazei de date.

Observam ca deocamdata cele doua tabele sunt complet independente si nu exista nimic comun acestora.

In cazul nostru avem de-a face cu o relatie clasica, ce se numeste '*1 la n*', sau '*one to many*'. Trebuie sa avem doua tabele, unul de tip **Parent** (ANGAJATI) si unul de tip **Child** (COPII). Intr-o relatie '*1 la n*', unei linii din tabela **Parent** ii pot corespunde mai multe linii din tabela **Child**. Pentru a rezolva aceasta relatie, in tabela **Child** (COPII) trebuie sa introducem o noua coloana care sa stocheze ID-ul corespunzator parintelui.

In tabela **COPII** mergeti in modul **Design View**.

Adaugam o noua coloana imediat dupa coloana **ID_COPII**: facem click dreapta pe linia **Prenume** si alegem **Insert rows**.



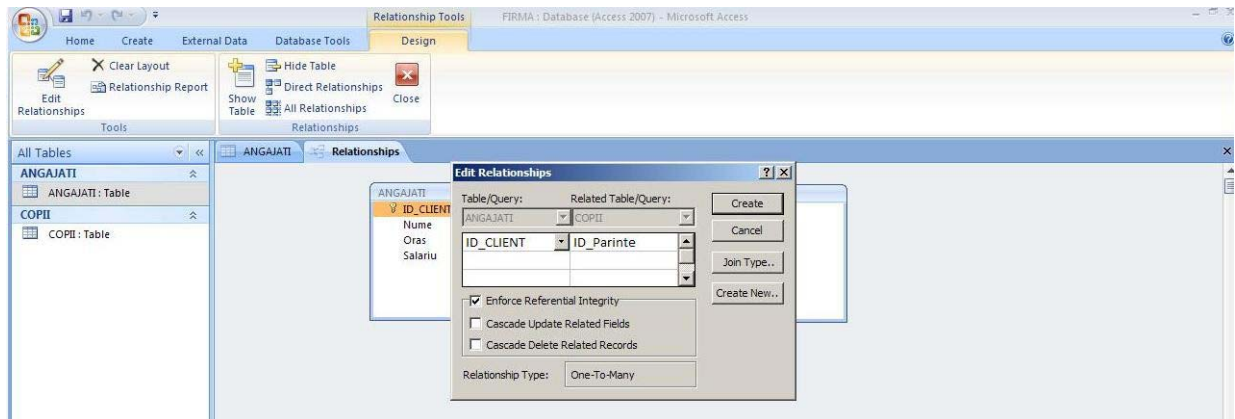
Introducem numele **ID_Parinte** si ca tip alegem acelasi ca si Primary Key-ul din **ANGAJATI** – **ID_CLIENT (Number)**. Salvam si revenim in modul **Datasheet View**.

Pentru a stabili exact relatia, adica pentru a specifica faptul ca datele din clooana **ID_CLIENT** din **ANGAJATI** este replicata in **ID_Parinte** din **COPII**, mergem in **Database Tools > Relationships**.



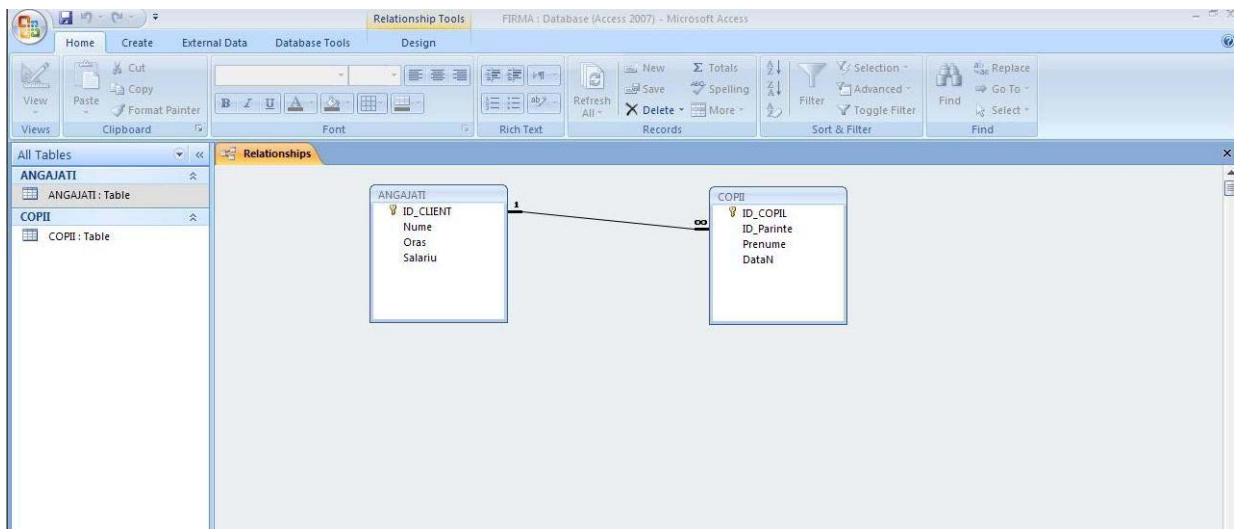
Selectam ambele tabele si apasam **Add**, apoi **Close**.

Cele doua tabele nu sunt inca relationate. Tragem (*Drag&Drop*) coloana **ID_CLIENT** din **ANGAJATI** peste coloana **ID_Parinte** din **COPII**. In fereastra deschisa bifam checkbox-ul **Enforce Referential Integrity**. Apasam **Create**.



Atentie! Este nevoie a inchidem tabelele inainte de a face aceasta operatiune de relationare.

Observam ca apare relatia '1 la n'. Salvam relatia.



Deschidem tabela *COPII*. Coloana *ID_Parinte* stocheaza ID-ul corezpunzator parintelui. De exemplu, daca in prima linie scriem 1, stabilim ca parintele lui *Gigel* este *Ionescu*, care are *ID_CLIENT* 1 in tabela de *ANGAJATI*.

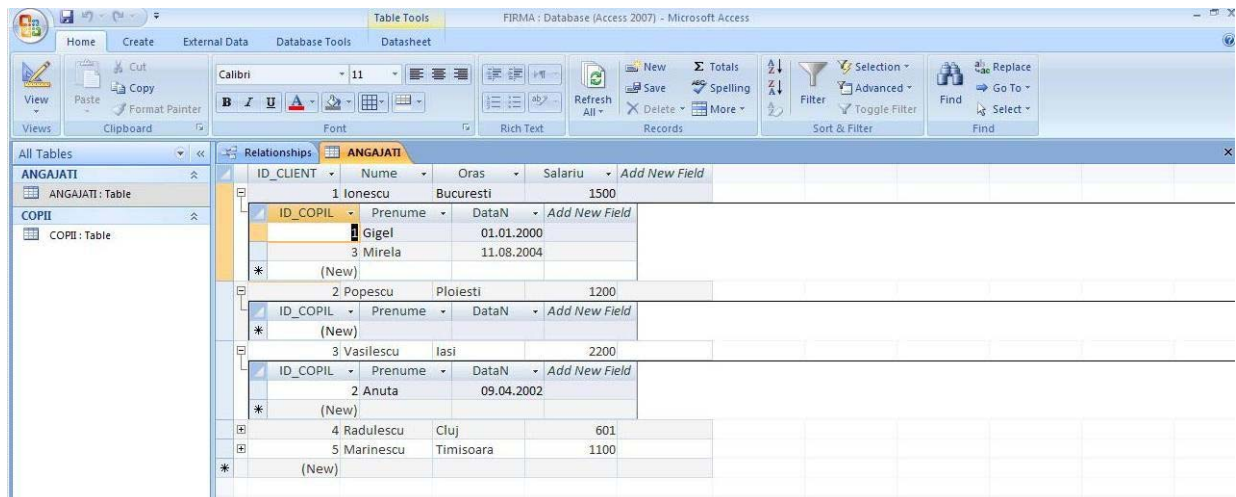


Tot in coloana *ID_Parinte* introducem 1 si pentru linia 3 (*Mirela*).



Aceasta inseamna ca parintele *Ionescu* are doi copii: *Mirela* si *Gigel*.

Introducem 3 in coloana **ID_Parinte** din randul 2 (*Vasilescu* are un singur copil, *Anuta*).



Mergem in tabela **ANGAJATI (Parent)**. **Access** ne permite sa vizualizam in mod direct (atata timp cat am stabilit relatia descrisa mai sus), pentru fiecare linie din tabela **parent**, care sunt liniile corespondente din tabela **child**. Interogarile sunt **date** extrase din baza de date. Este una dintre cele mai importante operatii dintr-o baza de date.

Crearea unei interogari

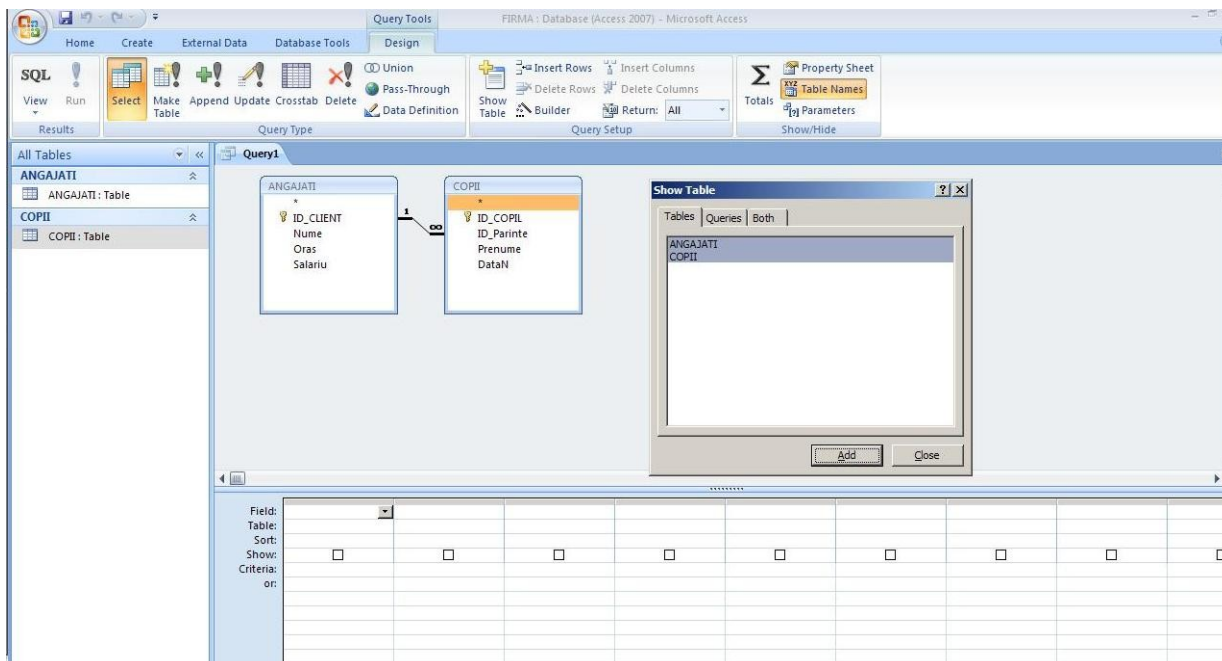
Avem doua tabele: *ANGAJATI* si *COPII*. Cele doua tabele sunt relationate: unei linii din tabela *ANGAJATI* ii pot corespunde mai multe linii din tabela *COPII*. Se poate observa in tabelul *ANGAJATI* ca pentru *Ionescu* avem doua inregistrari in tabela *COPII*.

Una dintre cele mai puternice operatii in **Access** este aceea prin care reusim sa interogam diferite obiecte. De exemplu, putem sa interogam ambele tabele in acelasi timp si sa aflam care este fiecare angajat, si care este copilul fiecarui angajat.

Din ribbon-ul **Create** alegeti **Query Design**.

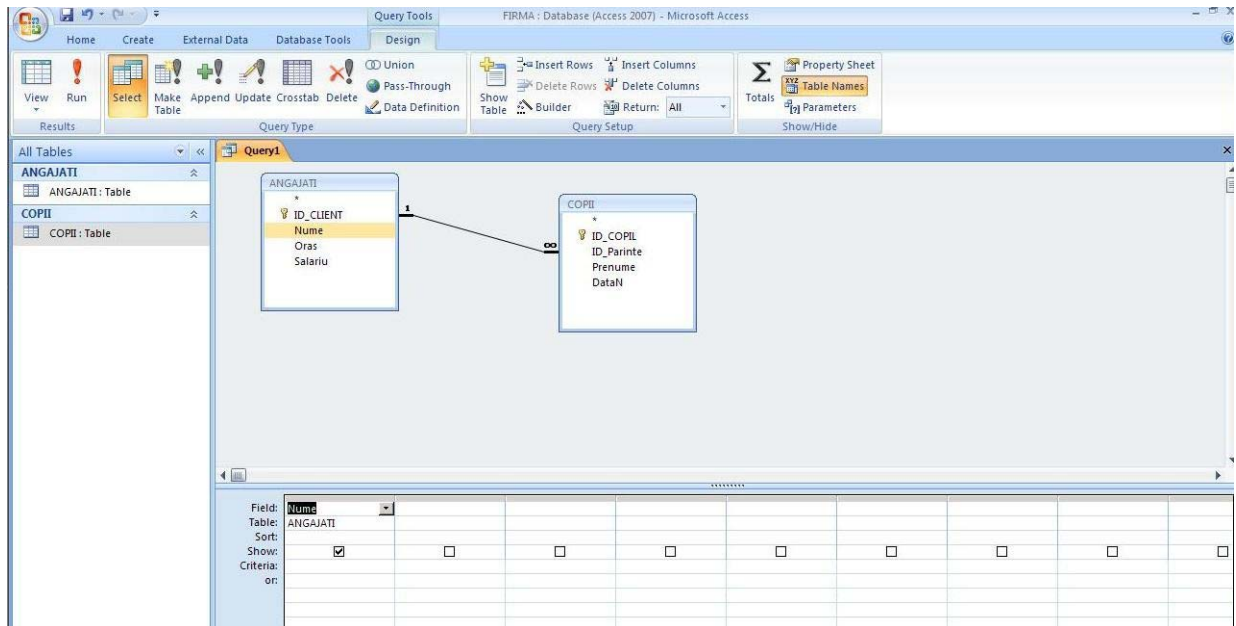


Alegem ambele tabele, apasam **Add**, apoi **Close**.

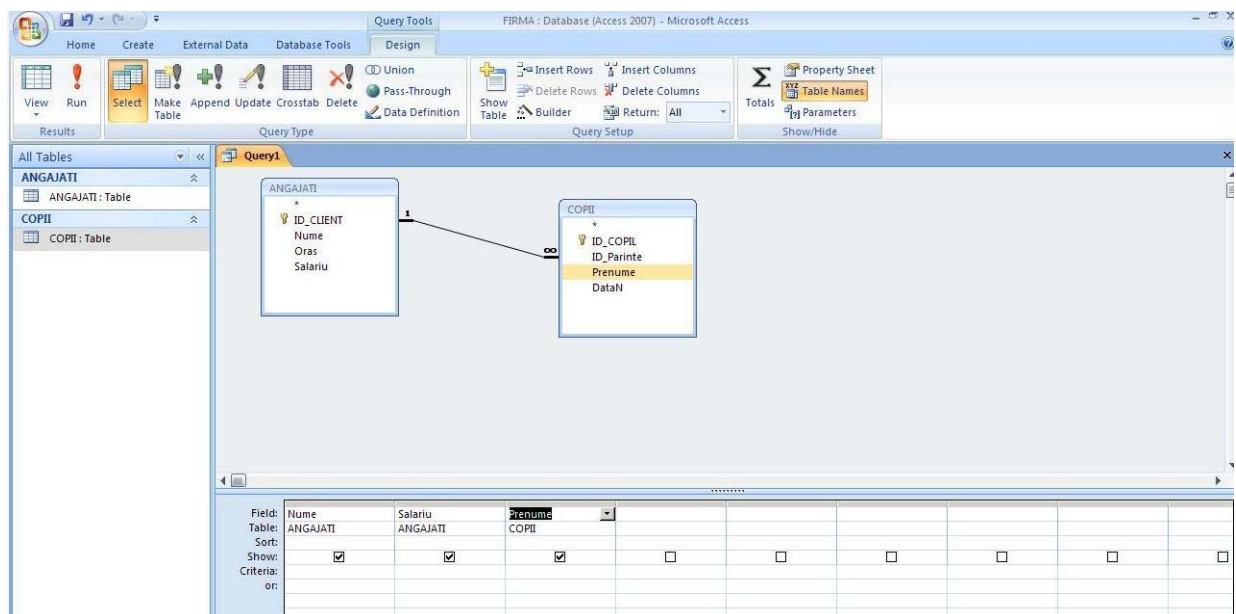


Observam ca este evidentiata si relatia dintre cele doua tabele. In acest moment putem sa vedem orice fel de informatie stocata in aceste tabele.

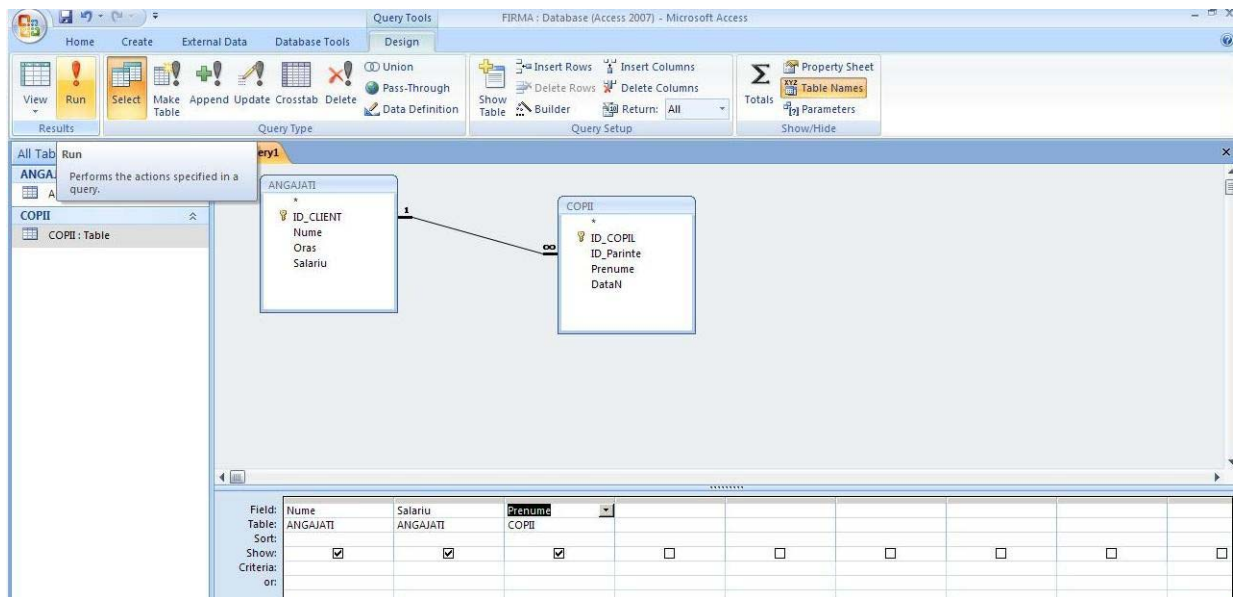
Facem dublu-click pe coloana *Nume* din tabela *ANGAJATI*, si ea este introdusa automat in acesta interogare.



Mai adaugam *Salariu* din tabela *ANGAJATI* si *Prenume* din tabela *COPII*.



Apasam **Run**, din ribbon-ul contextual **Design** > **Results**.



Se vor afisa inregistrarile gasite.

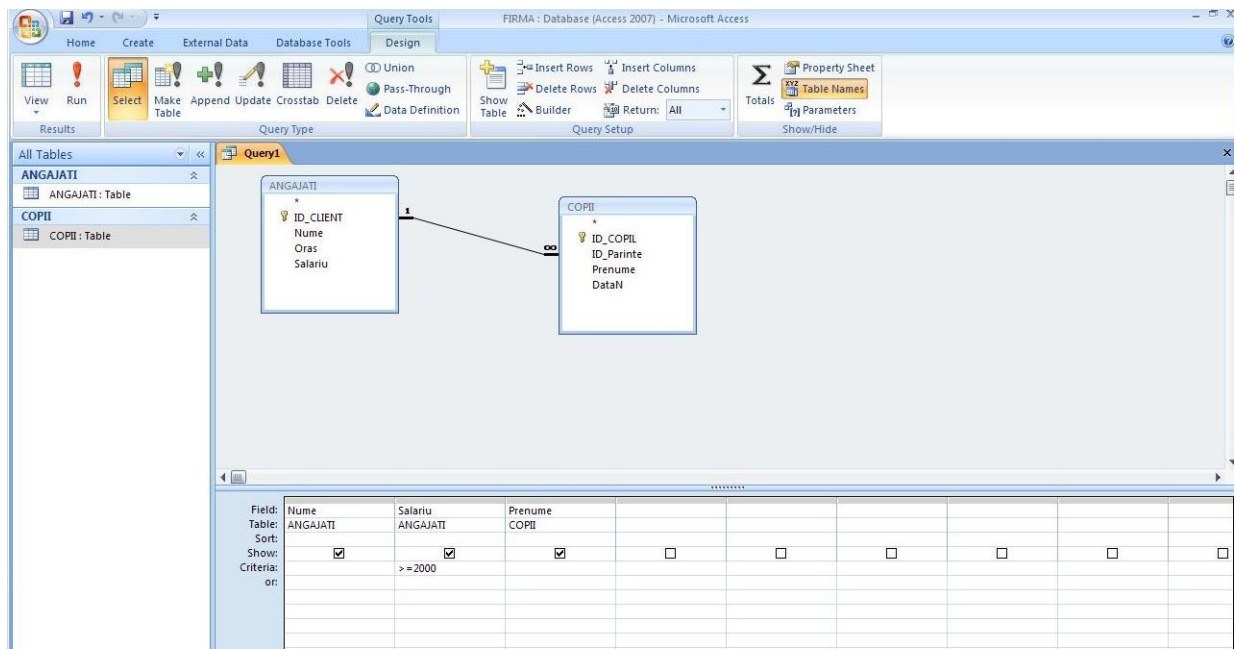


Observam ca informatiile despre *Ionescu* se repeta, pentru ca el are doi copii.

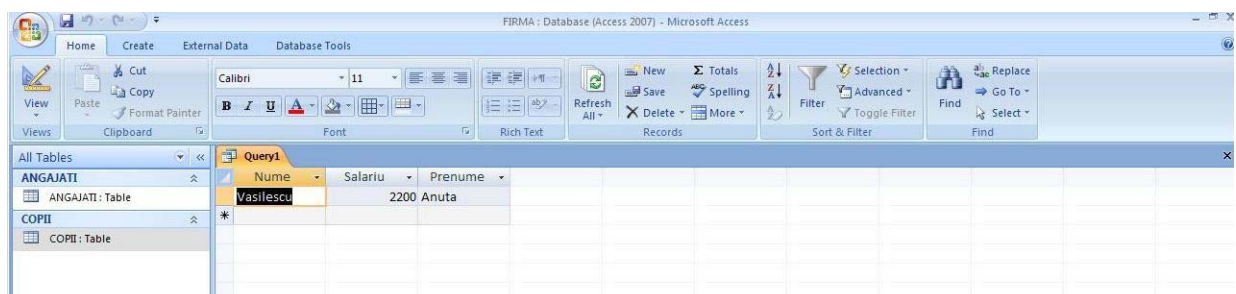
Ne intoarcem in **Design View**.

Ce este interesant, si aici vedem cu adevarat puterea interogarilor, este ca vom putea sa stabilim orice conditii logice dorim, vom putea sa filtram tabelele din **Access** stabilind niste conditii logice.

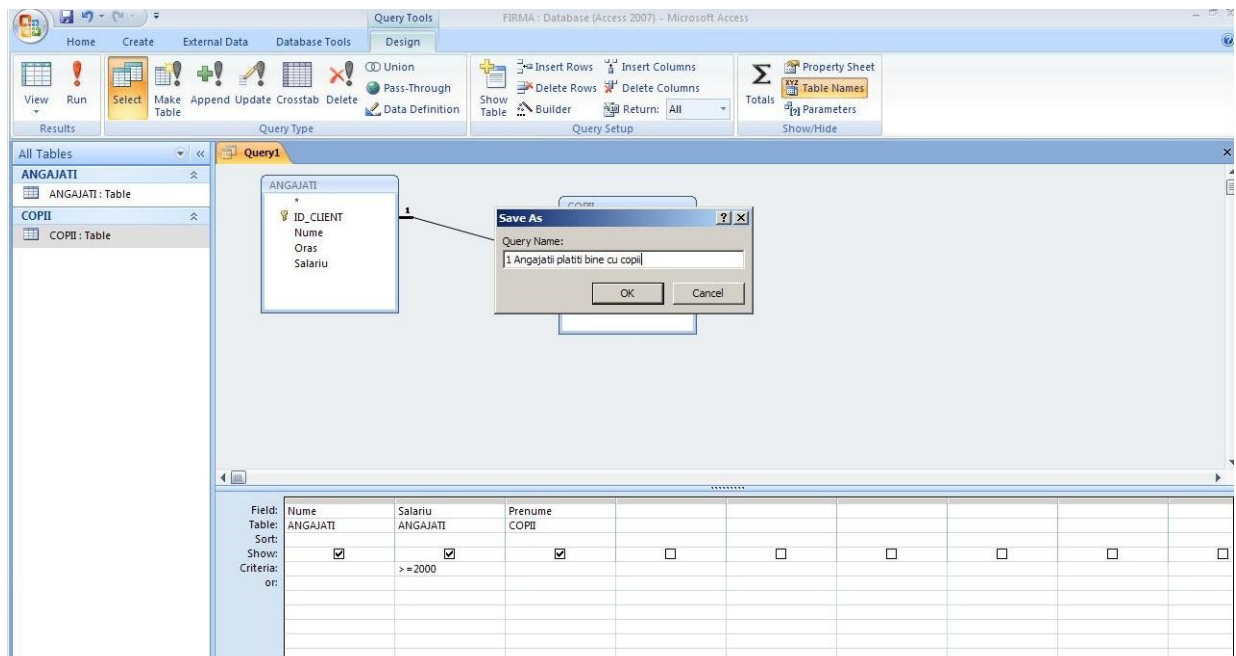
De exemplu, la angajati nu vrem sa vedem decat acei angajati care au un salariu mai mare de 2000. In linia *Criteria*, corezpunzatoare coloanei *Salariu* din interogare, introducem conditia: '>=2000'.



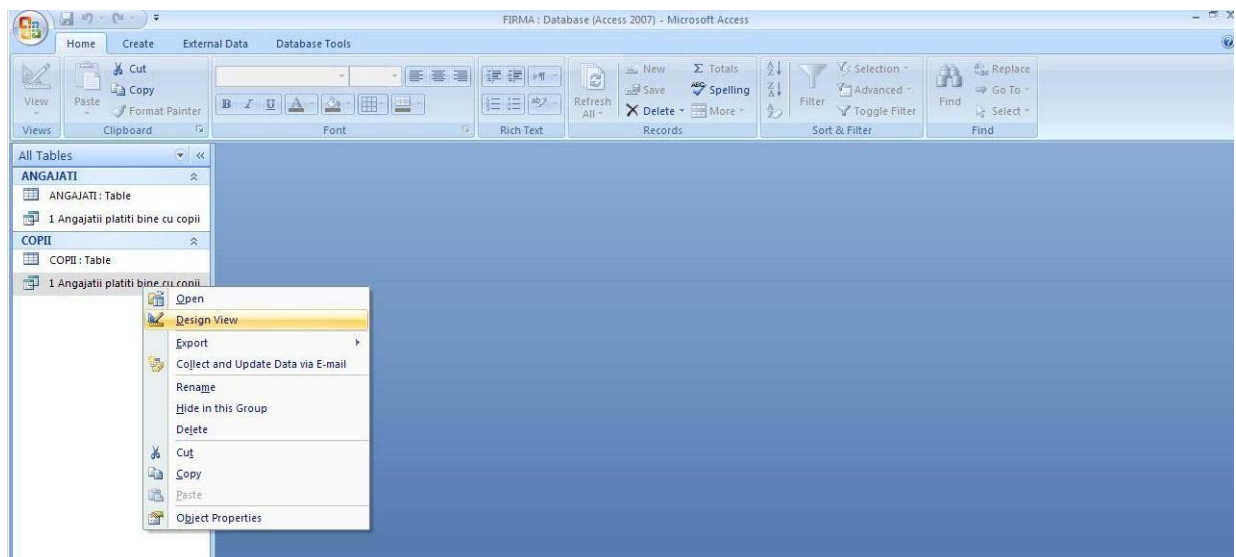
Apasam din noi **Run** si observam ca singurul angajat care are salariul mai mare de 2000, si are si copii, este *Vasilescu*.



Ne intoarcem in modul **Design View**. In acest moment putem sa salvam aceasta interogare pentru a o refolosi mai tarziu. Din **Quick Access Toolbar** apasam **Save** si introducem numele interogarii: '*1 Angajatii platiti bine cu copii*'. Apasam **OK**.



Inchidem designerul de interogare, si putem observa ca in acest moment, in meniul din partea stanga a ecranului avem doua tabele si o interogare cu numele mentionat mai sus. Oricand interogarea poate fi modificata facand click dreapta pe ea si alegand **Design View**.

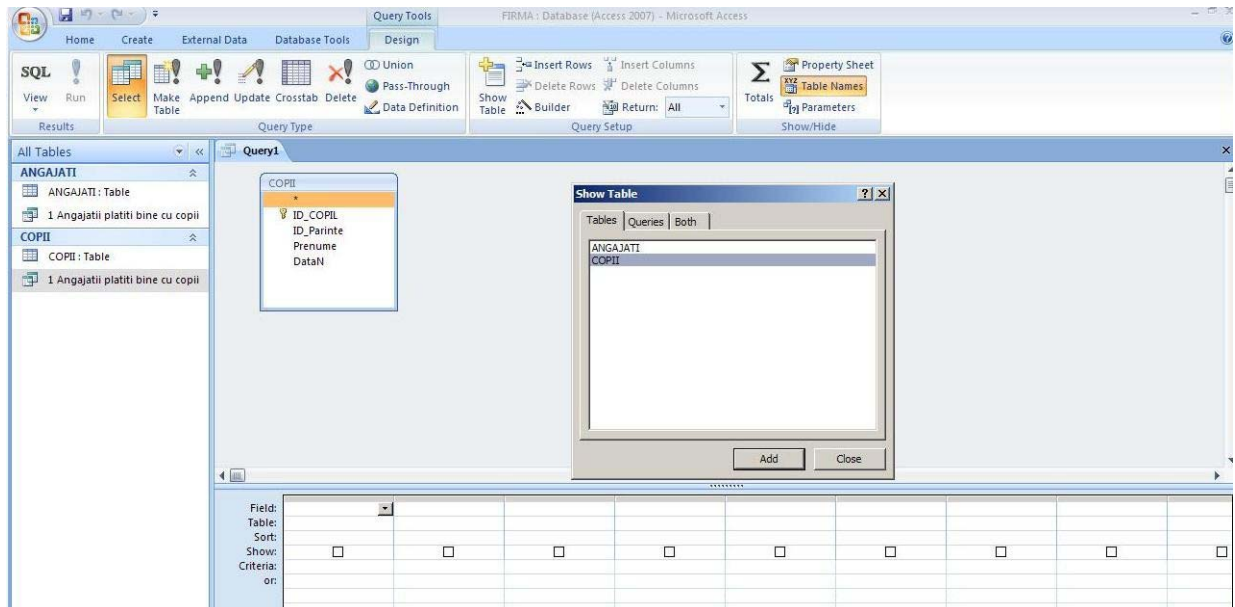


Interogările sunt **date** extrase din baza de date. Este una dintre cele mai importante operații dintr-o bază de date.

In baza noastră de date mai facem o interogare: toți copiii născuți după o anumită dată (anul 2001).

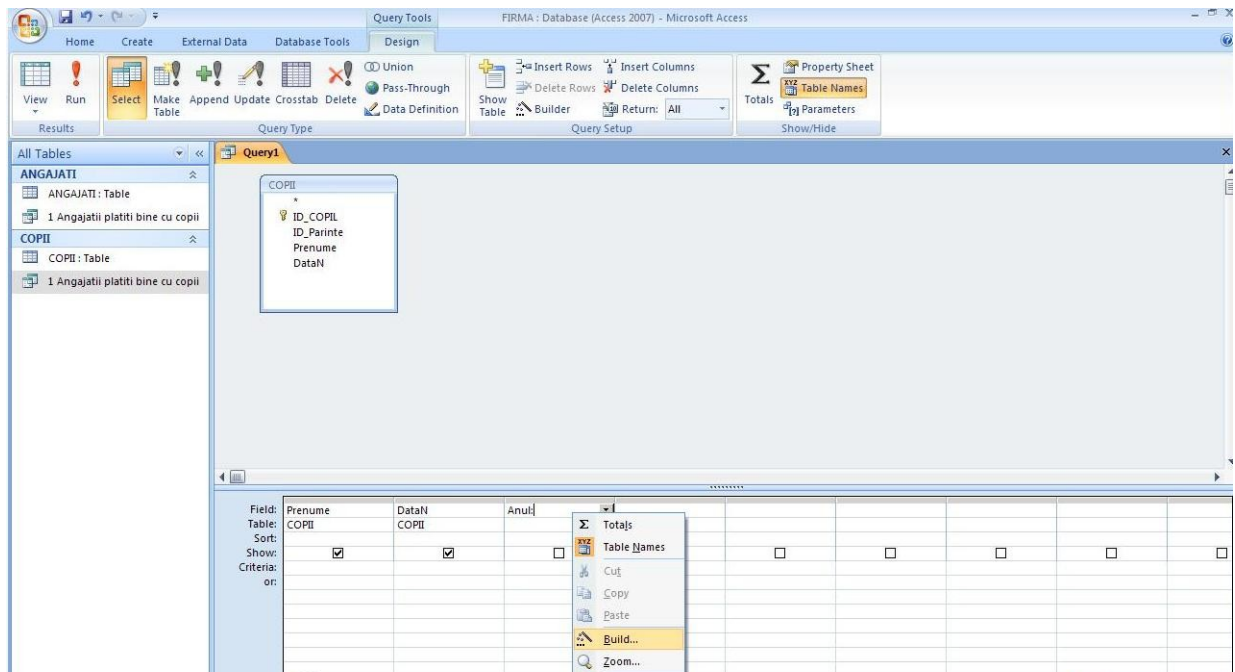
Inchidem tabelele, iar din ribbon-ul **Create** alegem să facem o nouă interogare – **Query Design**.

Alegem doar tabela *COPII*, și apăsăm **Add** apoi **Close**.



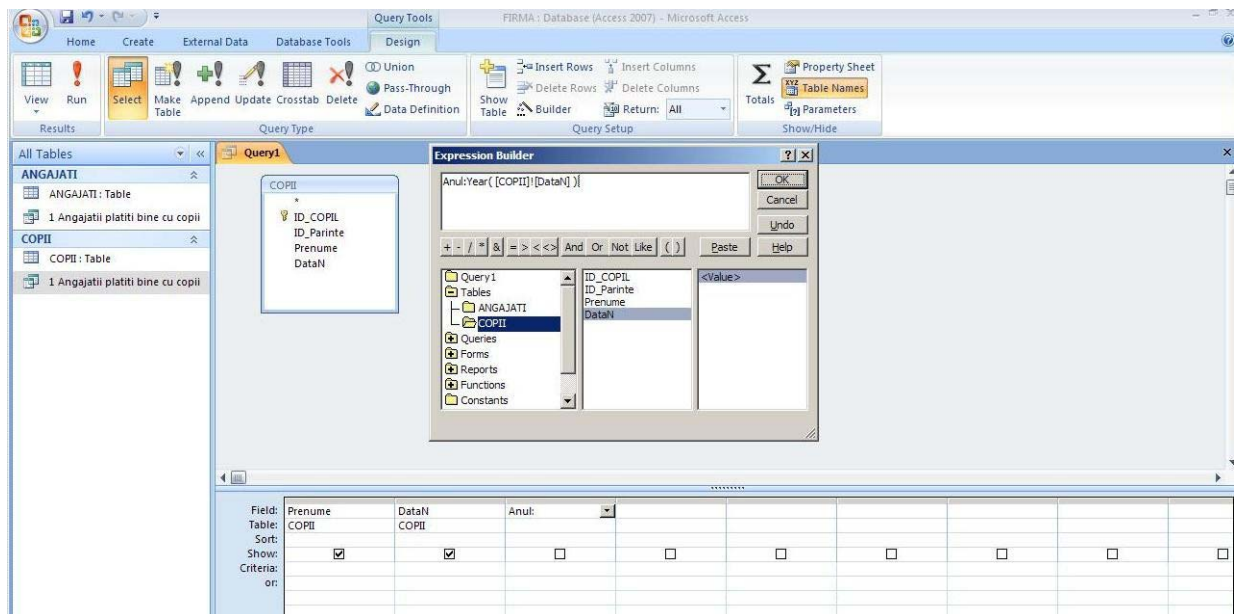
Daca selectam steluta care sa gaseste in caseta tabelului deasupra primei coloana, si apasam **Run** interogarea va aduce tabelul in intregime.

Folosim pentru interogare coloanele *Prenume* si *DataN*. Putem sa adaugam inca o coloana (coloana calculata), pentru care adaugam (in *Fields*) aliasul *Anul*. Facem click dreapta pe campul respectiv si alegem **Build**.



In fereastra nou deschisa (**Expression builder**) putem introduce formula de calcul. Asa cum in **Excel** exista functii, si in **Access** exista, dar acestea din urma nu sunt toate identice cu cele din **Excel**.

Daca vrem sa calculam anul de nastere al unui copil, folosim functia **Year**. Ca atribut alegem **Tables > COPII > DataN**, folosind designer-ul care ne permite sa navigam intre coloanele din tabelele create. Formula va devenii: **Anul: Year([COPII].[DataN])**. Apasam **OK**.

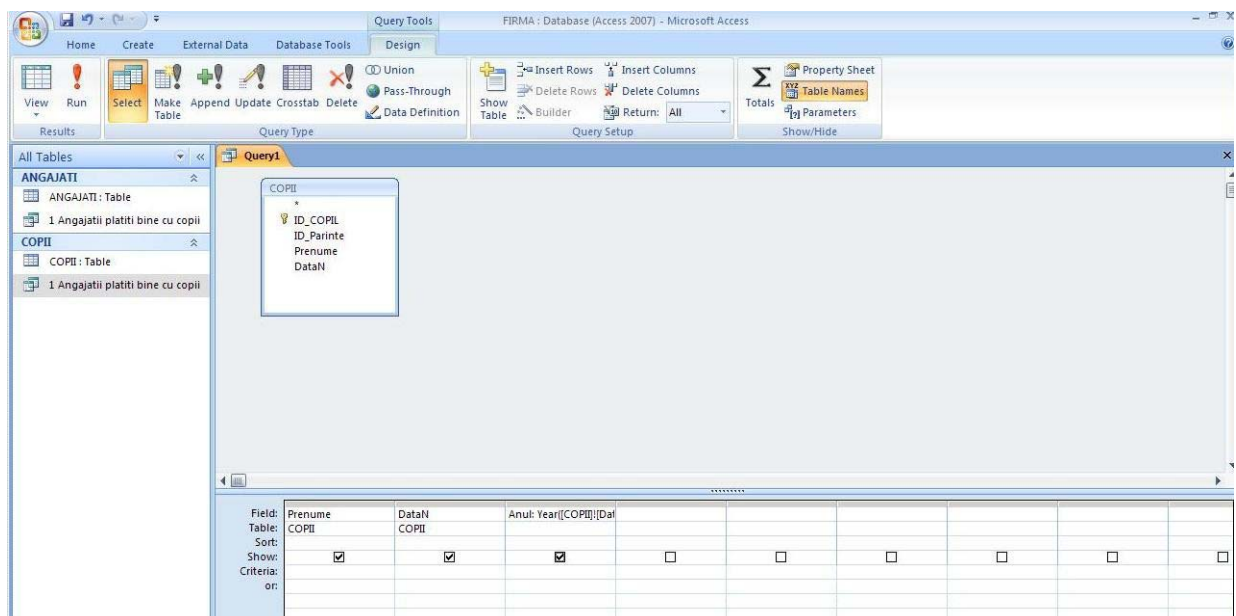


Bifam checkbox-ul **Show**, corespunzator acestei noi coloane si apasam **Run**.

Observam ca datele din coloana *DataN* este doar copiata din baza de date, iar coloana *Anul* este o coloana personalizata, care are in spate o formula.

In acest moment ne intoarcem la **Design View** pentru a stabili un criteriu de regasire, astfel ca afisam doar acei copii care au anul de nastere mai mare de 2001.

Pentru coloana *Anul*, in linia **Criteria** introducem conditia: '>=2001'.



Rulam din nou interogarea apasand **Run**. Observam ca numai doi copii sunt nascuti dupa anul 2001.

Scopul unui raport: **sa prezinte datele intr-un mod care sa permita interpretarea cat mai facila**

Un raport reprezinta o situatie extrasa dintr-o baza de date (o interogare) la care conteaza foarte mult:

- partea estetica (culori, tabele, border-uri, antet, subsol, conditional formatting, etc)
- modalitatea logica de afisare a datelor (grupari, sortari, etc).

Crearea unui raport

Un raport nu este altceva decat o interogare dintr-o baza de date ce urmeaza a fi interpretata de o anumita persoana. Acea persoana doreste sa vizualizeze repectivele date intr-un mod cat mai placut astfel incat sa poata cat mai usor sa extraga o informatie relevanta.

La rapoarte conteaza foarte mult doua aspecte:

- **Partea estetica (layout):** culori, border-uri, tabele;
- **Forma in care sunt prezentate datele:** antet, subsol, conditional formatting.

Avem deja doua interogari salvate.

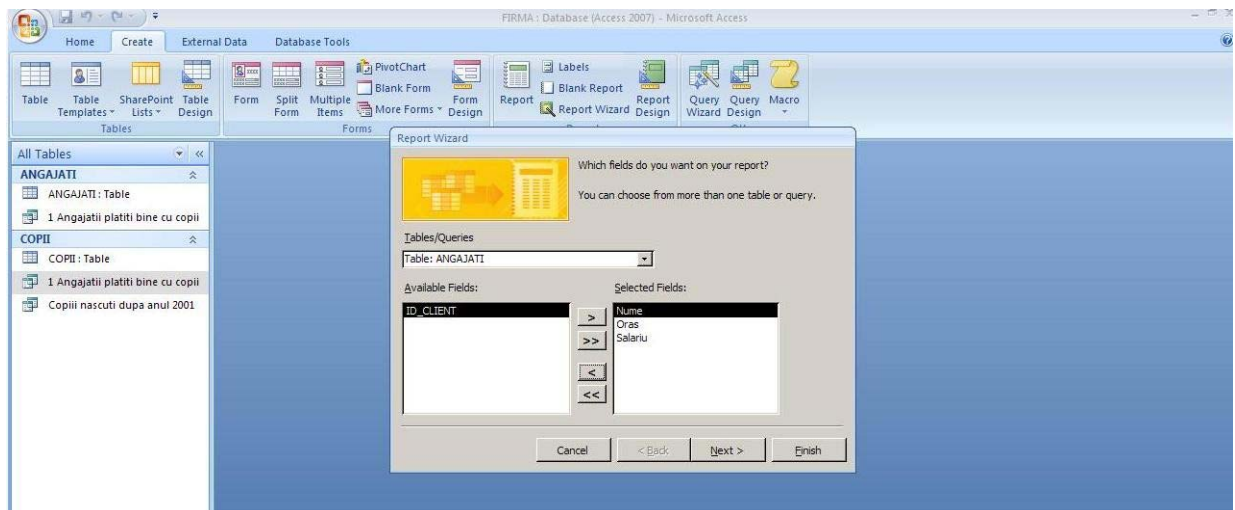
Pentru a face un raport, mergem in ribbon-ul **Create** si alegem **Report Wizard**.



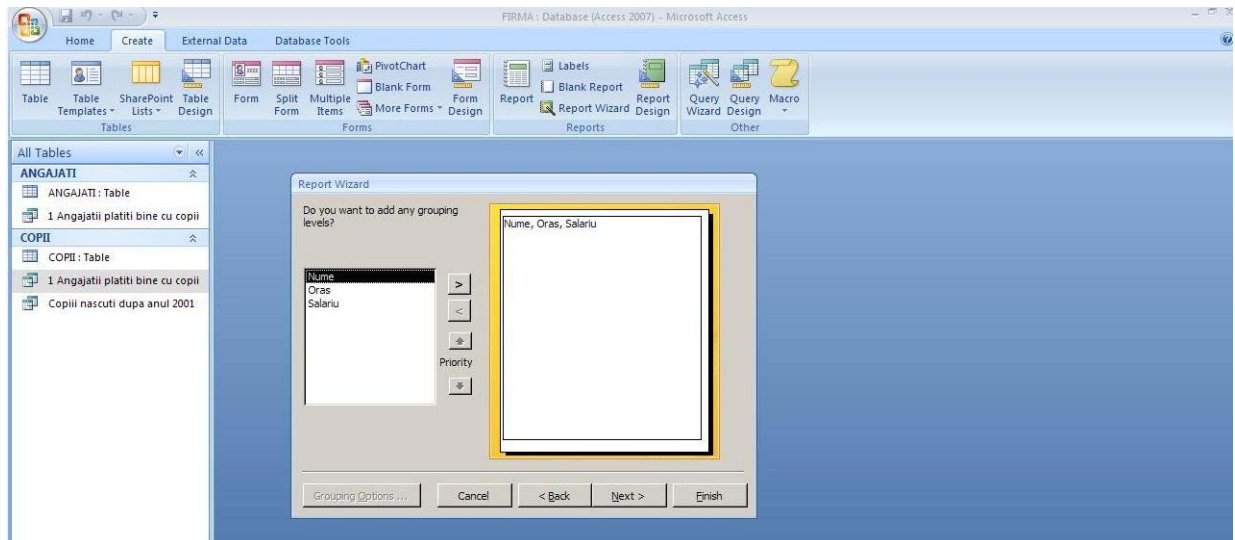
Vrajitorul (**Wizard**) ne intreaba ce sta la baza raportului nostru. Putem alege dintre tabelele si interogările deja create. Alegem tabela **ANGAJATI**.



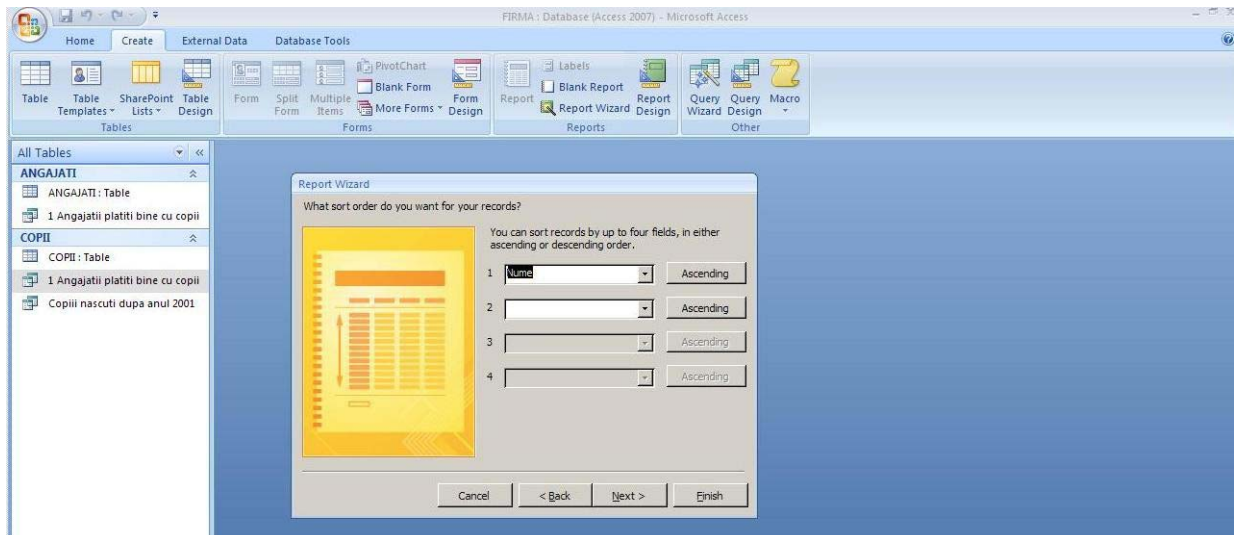
Acum putem sa selectam coloanale de care avem nevoie in raport. Alegem **Nume**, **Oras** si **Salariu**. Apasam **Next**.



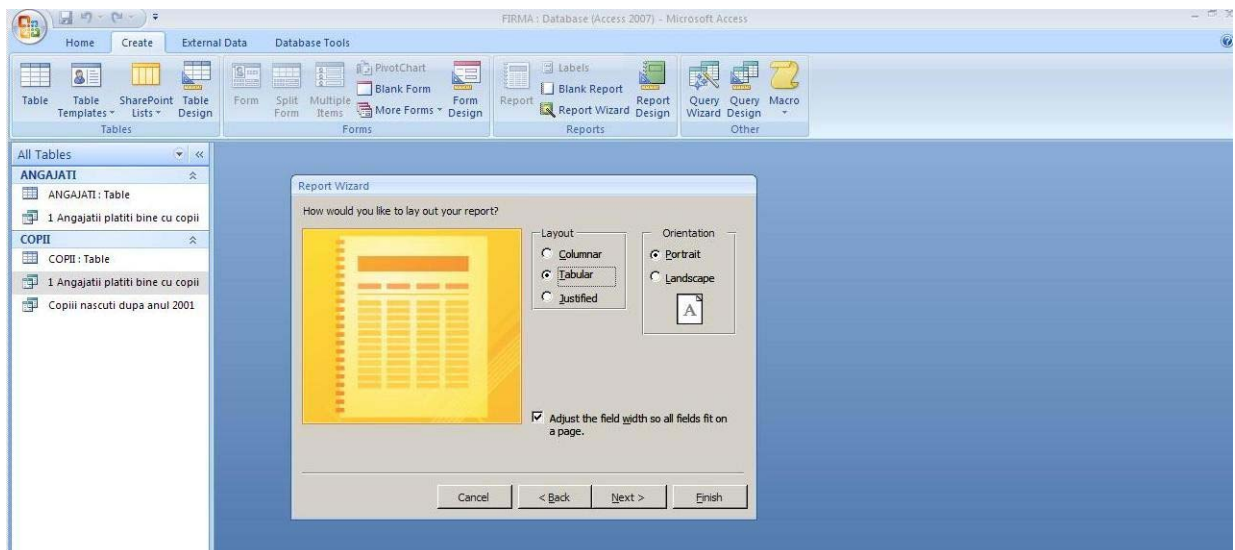
Putem apoi sa grupam datele intr-un anumit fel, dar nu avem nevoie asa ca apasam **Next**.



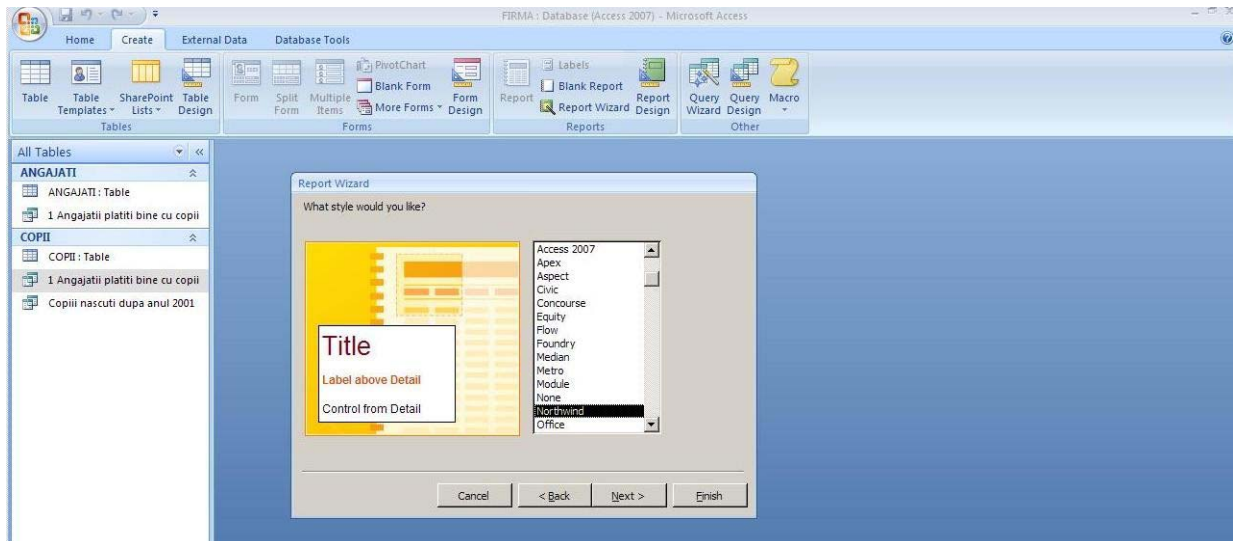
Putem de asemenea sa mentionam modul de sortare. Vom alege sa sortam dupa *Nume*, crescator. Apasam **Next**.



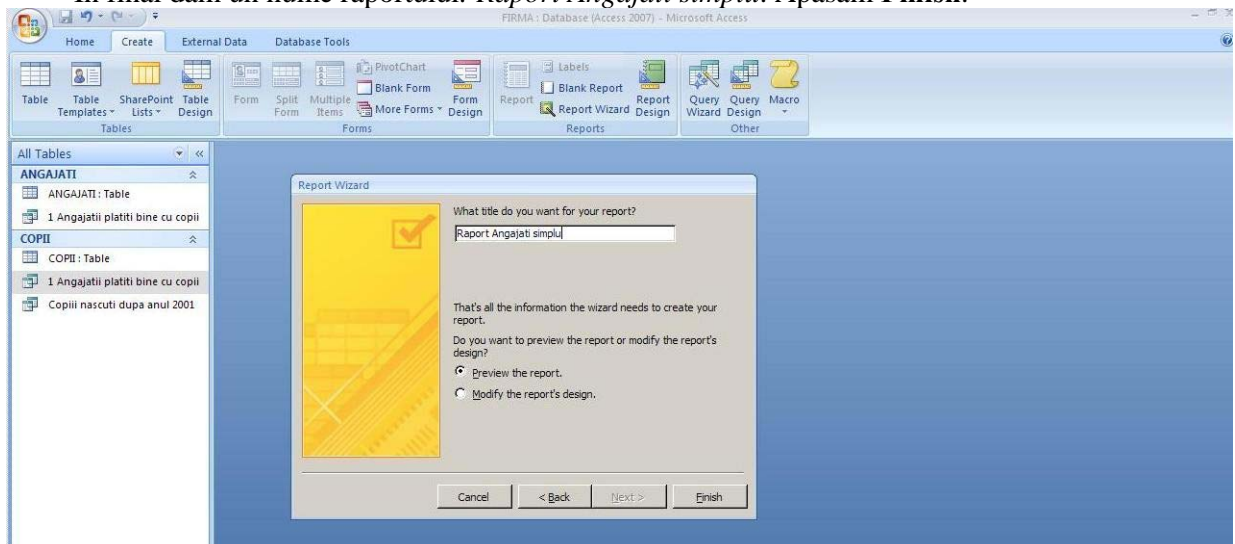
Putem sa alegem layout-ul raportului. Vom bifa **Tabular** (unul dintre cele mai comune) impreuna cu orientarea paginii **Portrait**.



Alegem un template pentru raportul nostru. Avem in lista o serie de stiluri predefinite care contin anumite atribute de formatare. Alegem **Northwind** si apasam **Next**.



In final dam un nume raportului: *Raport Angajati simplu*. Apasam **Finish**.



Observati ca raportul este tabela *ANGAJATI*, dar aspectul fizic, estetic, este diferit de cel din baza de date.

Scopul unui raport: **sa prezinte datele intr-un mod care sa permita interpretarea cat mai facila**

Un raport reprezinta o situatie extrasa dintr-o baza de date (o interogare) la care conteaza foarte mult:

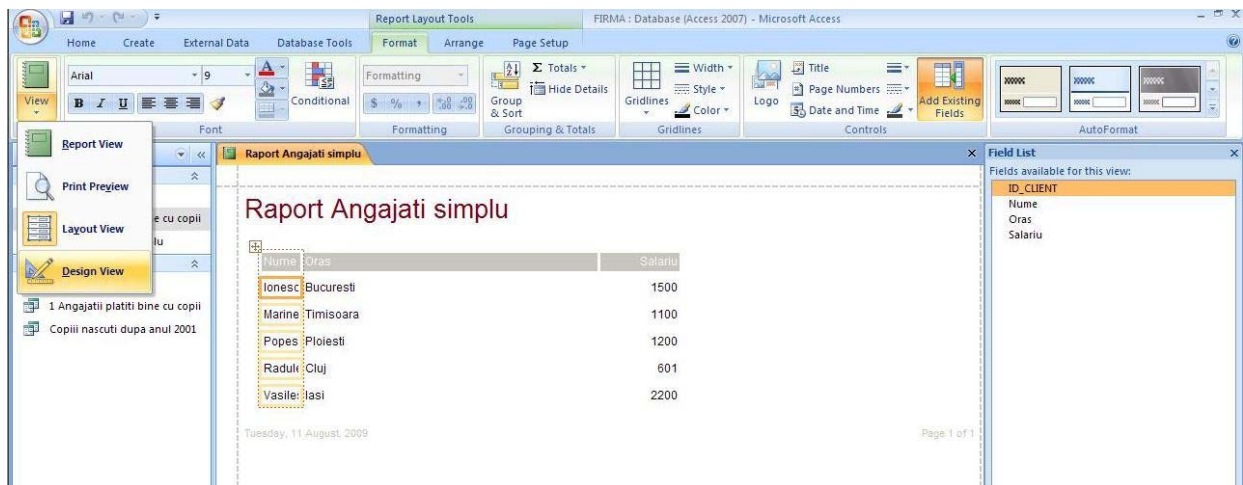
- partea estetica (culori, tabele, border-uri, antet, subsol, conditional formatting, etc)|
- modalitatea logica de afisare a datelor (grupari, sortari, etc)

Crearea unui raport. Formatarea Conditionata

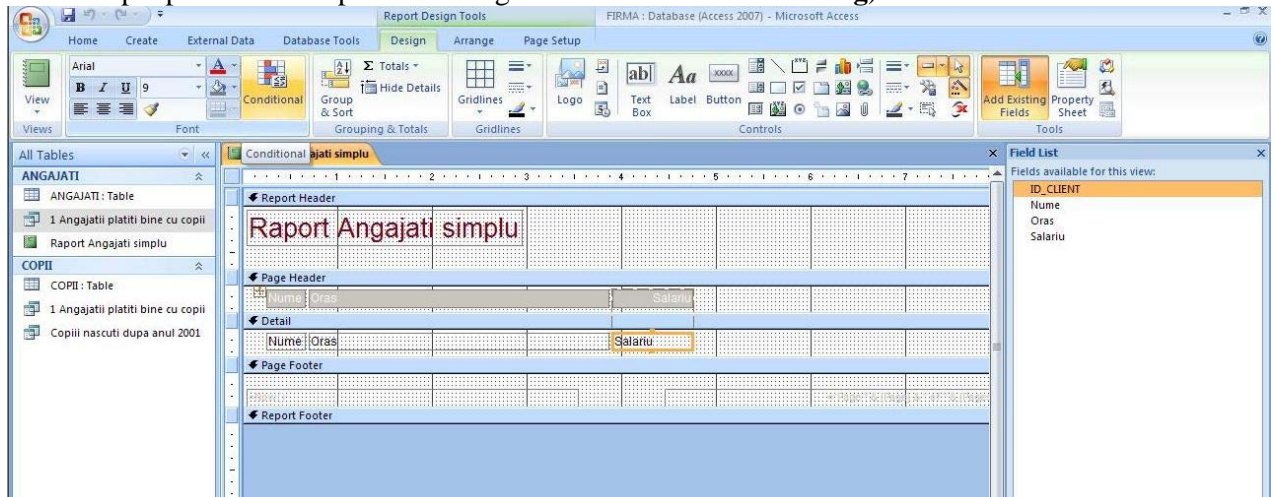
Un raport nu reprezinta un simplu *query* (interogare) cu un cap de tabel, cu un titlu, ce urmeaza a fi scos la imprimanta. Un raport de regula implementeaza operatii mult mai serioase, mult mai complexe. Vom aborda una dintre cele mai spectaculoase astfel de operatiuni: **formatarea conditionata**.

Am putea astfel sa impunem o conditie logica pe o anumita coloana astfel incat acele valori care indeplinesc acea conditie sa fie scoase in evidenta intr-un anumit fel. De exemplu, toti angajatii care au salariul mai mare de 1400 sa fie scrisi cu rosu.

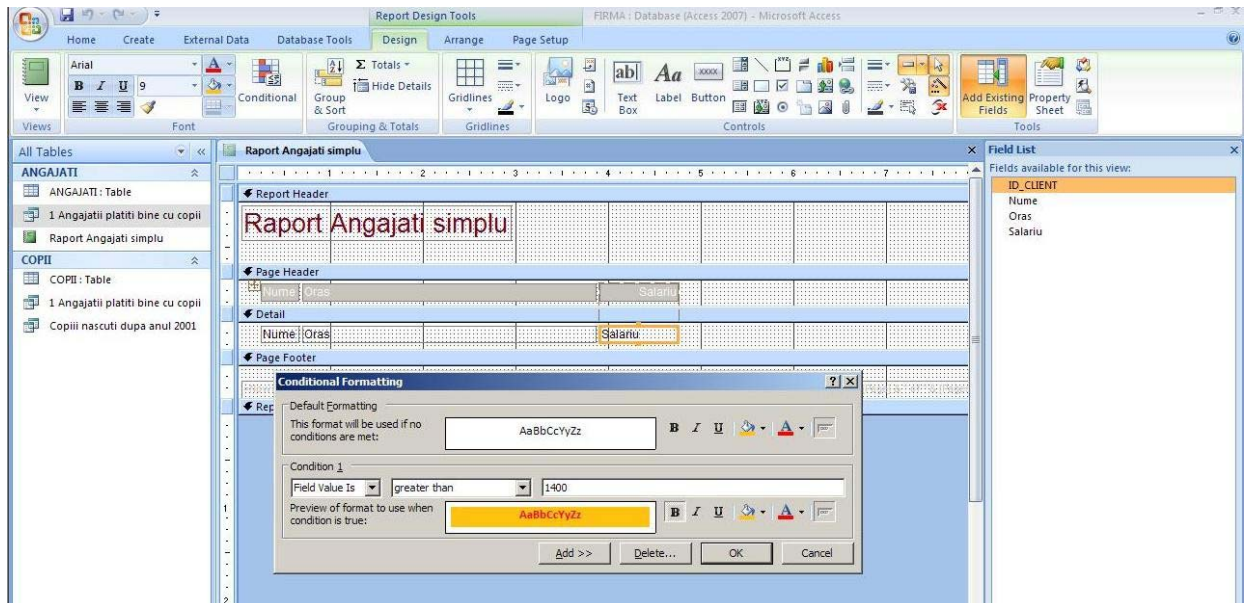
Mergem in modul de vizualizare **Design View** al raportului '*Raport Angajati simplu*'.



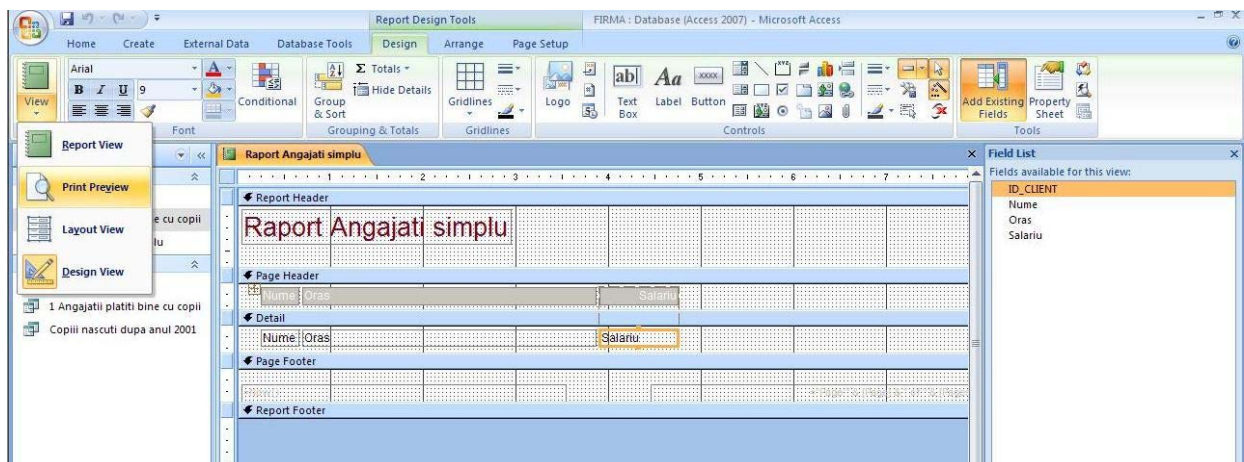
Selectam coloana *Salariu*, cea pentru care vrem sa aplicam formatarea conditionata, iar de pe ribbon-ul principal facem click pe **Conditional** (acelasi lucru il puteam obtine facand click dreapta pe coloana respectiva si alegand **Conditional Formatting**).



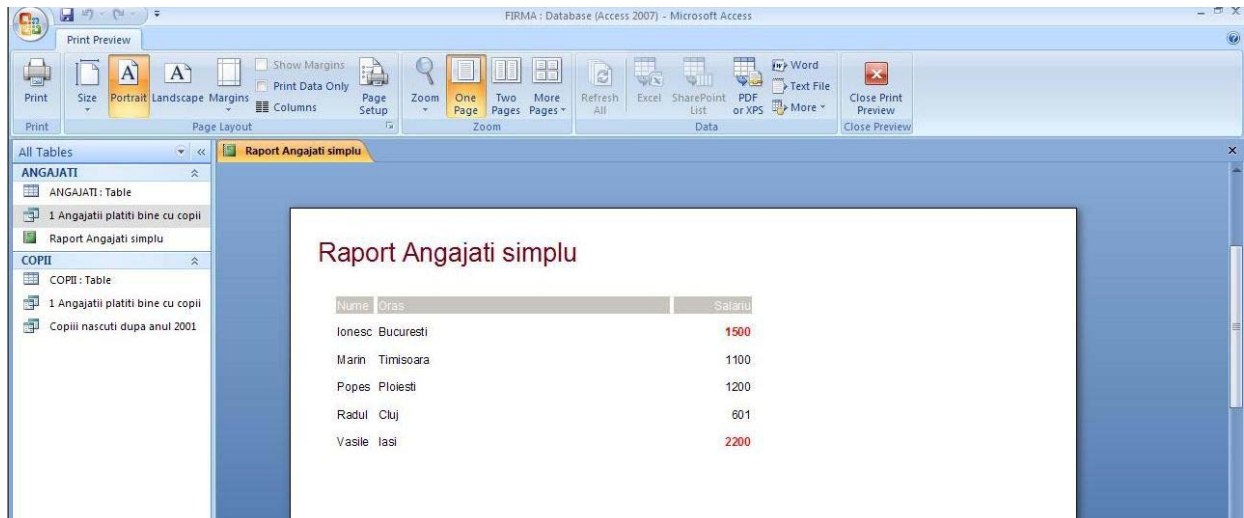
Cei care ati lucrat in **Excel** puteti recunoaste aceasta fereastra. Vom alege formatarea: **Field Value is, grather than**, apoi introducem valoarea *1400*. Din atributele de formatare alegem **Bold**, **Font Color** rosu si **Fill/BackColor** galben. Apasam **OK**.



Alegem modul de vizualizare **Print Preview**.



Observati ca toate acele valori care respecta conditia logica sunt scoase automat in evidenta.



Crearea unui formular

Formularele reprezinta ferestre ce permit adaugarea, modificarea, stergerea datelor dintr-o baza de date. Formularele sunt special create astfel incat sa permita cat ma facil si rapid operatiile de mai sus.

Formularele reprezinta obiecte din interiorul unei baze de date care ne ajuta sa modificam datele din tabele intr-un mod mult mai placut si mai usor.

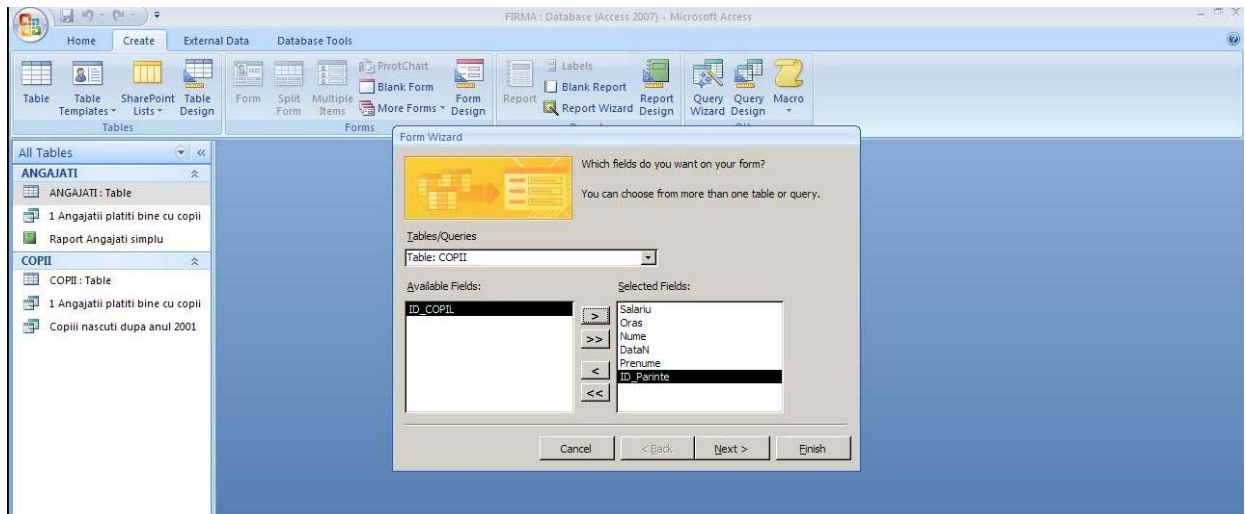
In mod normal, daca vrem sa modificam sau sa adaugam randuri noi in tabela *ANGAJATI* ar trebui sa o deschidem si sa operam modificarile in **Datasheet View**. Aceasta e o operatie destul de riscanta pentru ca exista posibilitatea sa alteram liniile din tabela si sa introducem valori neconforme cu realitatea.

Formularele exact acest lucru il fac: ne permit sa introducem date sau sa modificam, respectand anumite reguli. Formularele din **Access** (si formularele in general) sunt relativ complicat de realizat, de aceea pentru realizarea unor formulare profesioniste va trebui sa cunoasteti destul de multe despre **Access**.

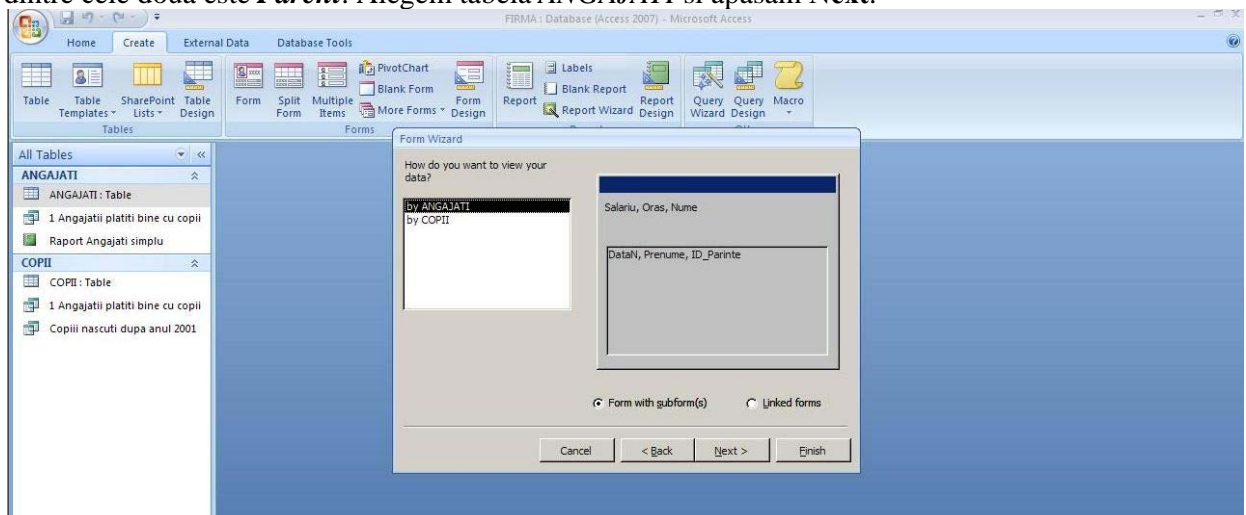
Din ribbon-ul **Create** alegeti **More Forms** > **Form Wizard** pentru crearea unui formular.



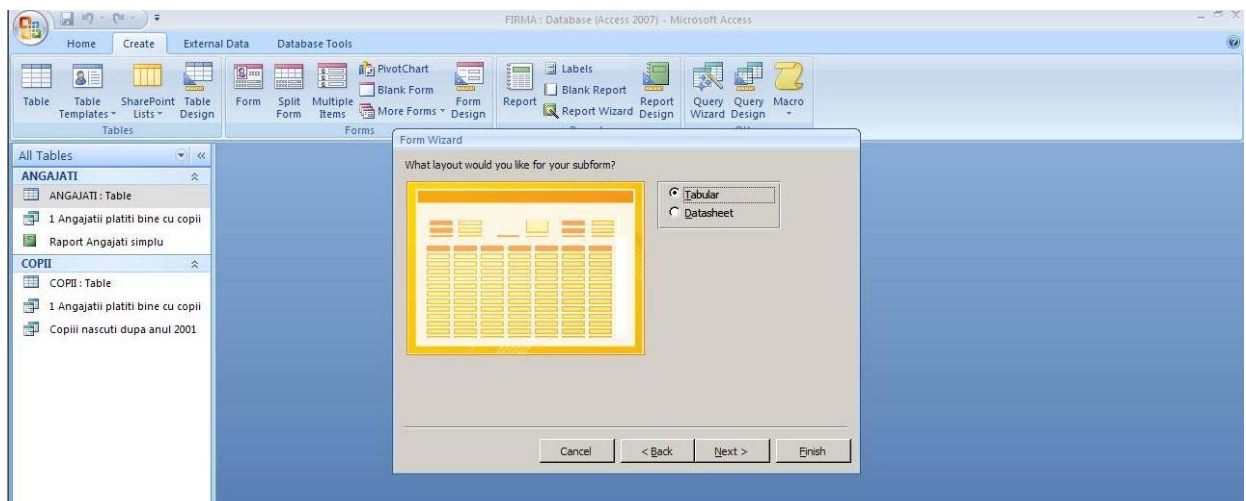
In fereastra de wizard deschida alegem campurile *Salariu*, *Oras* si *Nume* din tabela *ANGAJATI*, iar din tabela *COPII*, alegem *ID_Parinte*, *Prenume* si *DataN*. Vom putea sa modificam datele din doua tabele in acelasi timp. Apasam **Next**.



Access si-a dat seama ca am ales date din doua tabele si acum trebuie sa alegem care dintre cele doua este **Parent**. Alegem tabela **ANGAJATI** si apasam **Next**.



Ca layout alegem modul **Tabular**. Apasam **Next**.



Alegem un stil de formatare si apasam **Next**. In final, completam numele pentru formular si sub-formular:

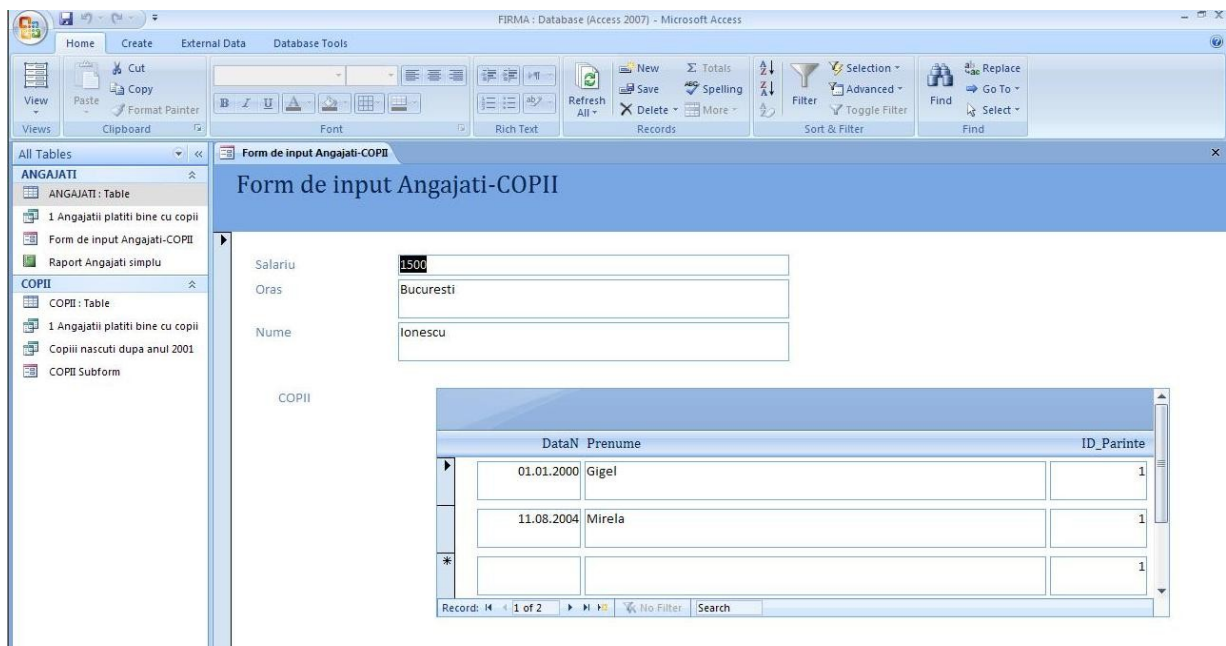
- *Form de input Angajati-COPII*
 - *COPII Subform*
- Apasam **Finish**.



Observati ca avem o singura fereasta de unde am putea modifica si angajati (partea de sus) si copiii angajatilor (partea de jos, in functie de parintele selectat).

Exista doua modalitati de navigare: navigarea de jos, care ne permite sa navigam intre liniile din *parent* – *ANGAJATI*, si navigarea din tabela *child* care ne permite sa navigam intre copii angajatului respectiv.

Daca vrem sa trecem la urmatorul angajat, apasam butonasul **Next Record**, din partea de jos.



Observam ca acum este afisat angajatul *Popescu* care nu are nici un copil.

Cu acest instrument putem sa facem orice operatie de modificare fara a mai fi nevoie sa deschidem ambele tabele.

Cu angajatul *Ionescu* afisat, apasam **New Record** in tabela *child*.

Form de input Angajati-COPII

Salariu: 1500
Oras: Bucuresti
Nume: Ionescu

COPII

DataN	Prenume	ID_Parinte
01.01.2000	Gigel	1
11.08.2004	Mirela	1
		1

Record: 1 of 2

Introducem prenumele *Ionica* si data nasterii.

Form de input Angajati-COPII

Salariu: 1500
Oras: Bucuresti
Nume: Ionescu

COPII

DataN	Prenume	ID_Parinte
01.01.2000	Gigel	1
11.08.2004	Mirela	1
03.08.2006	Ionica	1

Record: 3 of 3

In acest moment nu numai ca a fost introdus in tabel copilul *Ionica*, dar el a si fost asignat agajatului *Ionescu*. Daca deschidem tabela *COPII* observati noua inregistrare pentru *Ionica*, care are deja la *Parinte_ID* valoarea 1 (care il reprezinta pe angajatul *Ionescu*).

In acest formula nu se pot face numai adaugari de linii noi, dar si modificari, sortari, filtrari, s.a.m.d.

Cuprins

Introducere	1
Creare unei baze de date de la 0	1
Crearea unui tabel	4
Crearea constrangerilor	7
Crearea relatiilor	10
Crearea unei interogari	15
Crearea unui raport.....	23
Crearea unui raport. Formatarea Conditionata.....	27
Crearea unui formular.....	29
Cuprins	33